Deployment Related Mental Health Care Seeking Behaviors in the U.S. Military and the Use of Telehealth to Mitigate Their Impacts on Access to Care

by

John Hess

Bachelor of Science, Mechanical Engineering
University of Michigan, 2008

Submitted to the Engineering Systems Division and the Department of Aeronautics and Astronautics in Partial Fulfillment of the Requirements for the Degrees of

Master of Science in Technology and Policy and Master of Science in Aeronautics and Astronautics

at the

Massachusetts Institute of Technology

June 2012

Signature of Author ....
Technology and Policy Program, Engineering Systems Division
Department of Aeronautics and Astronautics
May 11, 2012

Certified by ....
Jayakanth Srinivasan
Research Scientist, Lean Advancement Initiative
Thesis Supervisor

Certified by ....
Deborah Nightingale
Professor of the Practice of Aeronautics and Astronautics and Engineering Systems
Co-Director, Lean Advancement Initiative
Director, Sociotechnical Systems Research Center
Thesis Supervisor

Accepted by ....
Joel P. Clark
Professor of Materials Systems and Engineering Systems
Acting Director, Technology & Policy Program

Accepted by ....
Eytan H. Modiano
Professor of Aeronautics and Astronautics
Chair, Graduate Program Committee

Deployment Related Mental Health Care Seeking Behaviors in the U.S. Military and the Use of Telehealth to Mitigate Their Impacts on Access to Care

by

John Hess

Submitted to the Engineering Systems Division and the Department of Aeronautics and Astronautics on May 11, 2012 in Partial Fulfillment of the Requirements for the Degree of Master of Science in Technology and Policy and Master of Science in Aeronautics and Astronautics

Abstract

Interviewees report that groups of service members returning from Iraq and Afghanistan often require substantial amounts of mental health care, causing surges in demand at military hospitals. These hospitals have difficulty keeping up with demand during the busiest periods. The exact patterns of demand during surges are difficult to measure because the military records utilization, but not actual need for services.

This thesis analyzes the care seeking behaviors of service members and their families across the deployment cycle using historical data. This analysis shows that service members and their families seek more care after each deployment. More importantly, it shows that service members seek care at higher rates in predictable intervals following their deployments. New patient arrival rates are projected for several installations by multiplying actual installation populations by newly calculated care seeking rates. These projections show deployment related care seeking behaviors generate surges in demand and thereby validate qualitative findings from field work.

A simulation of the military's system of care uses these demand projections to specify patient arrival patterns. Comparison of several simulated scenarios shows that surges make it very difficult for individual military hospitals to offer access to care using only their own mental health care providers. Allowing hospitals to share their providers with one another offers little improvement.

As hypothesized, using a group of dedicated telehealth providers to support the most overburdened installations can offer a substantial improvement in access to care. This insight leads to four policy recommendations. First, a service wide or joint scheduling system should be created. Second, telehealth can best support overburdened hospitals when some providers are dedicated solely to surge support. Third, the services should take responsibility for meeting access to care goals instead of delegating the burden to installations. Lastly, hiring actions should be tied directly to an accurate measurement of excess demand.

Thesis Supervisor: Jayakanth Srinivasan
Title: Research Scientist
Thesis Supervisor: Deborah Nightingale
Title: Professor of the Practice of Aeronautics and Astronautics and Engineering Systems

Acknowledgements

The people in my life have defined my time at MIT. My fellow student researchers have made my time here more enjoyable than any work has the right to be. JK, my advisor, has been a mentor and a friend.

More than anything, I'm grateful for the love and support of my family. Mom, Dad, and Fred, I love you.

Contents

List of Figures

Nomenclature

BCT Brigade Combat Team

DoD Department of Defense

EBH Embedded Behavioral Health

FY Fiscal Year

M2 MHS Mart

MEDCOM Army Medical Command

MHS Military Health System

MTF Military Treatment Facility (military hospital)

NDAA National Defense Authorization Act

PTSD Post-traumatic Stress Disorder

RVU Relative Value Unit

TH Telehealth

1 Introduction

1.1 Motivation

Military leaders and policy makers, up to and including the Chairman of the Joint Chiefs of Staff, have called for improvements across the continuum of mental health care for the men and women in their charge. Access to care is often a target of policy makers when attempting to improve healthcare systems, and the military's psychological health system is no exception.

Many military installations are either consistently understaffed or overwhelmed by surges in demand when large groups of servicemembers return from deployment. Scarce and delayed care exact a cost on individuals and on the military as a whole. Several reports since the invasions of Afghanistan and Iraq in 2001 and 2003 [Tanielian and Jaycox, 2008, Arthur et al., 2007, Hoge et al., 2004] identified access-to-care as a challenge for service members and for their families. Interviews with dozens of providers, leaders, servicemembers and advocates and visits to several sites over 2011 and 2012 identified access-to-care as a continued challenge. Virtually none of these interviewees suggested curtailing any efforts to prevent or treat psychological health conditions. But resources – especially human resources – are finite. Providers in rural areas, where many military installations are located, are difficult to recruit and retain [Tanielian and Jaycox, 2008].

One promising option for improving access is the use of telehealth to connect military providers with patients around the world. Today, coordinating care across many sites is difficult, so telehealth is only used when the high administrative overhead it requires is worthwhile. It is possible though, that new scheduling systems, human resources allocations and network topologies could improve access to care.

Results from this thesis should help researchers and practitioners better understand the patterns of demand generated by recently deployed service members and their families. These results should also help policy makers create a telehealth system that provides the best return on investment of their political capital and the military health care system's human resources.

1.2 Research Questions

This analysis addresses three related research questions:

1. How do the care seeking behaviors of service members and their families change over the course of deployments?
2. Do surges in demand at military hospitals significantly decrease access to care?
3. Can telehealth help the military to deliver better access to care through load balancing?

1.3 Thesis Outline

Section 2 provides an overview of mental health in the military, access to care, military mental health care policy and models similar to the one presented in this thesis. Section 3 defines the scope of this analysis.

Section 4 shows the deployment related care seeking patterns of service members and their families in each service. Using these patterns, it projects demand for a set of Army and Marine installations. Section 6 and Section 7 use demand projections for six Army installations and the discrete event simulation presented in Section 5 to evaluate a several different telehealth arrangements.

Section 6 shows that the combination of care seeking behaviors and deployment patterns at Army power projection platforms causes large surges in demand and that these surges put significantly more stress on the system of care than constant arrivals at the same average rate would.

Having seen that these patterns create variable demand for individual installations, Section 7 shows that telehealth is especially useful for improving access to care when patients can be frictionlessly referred to and seen by distant providers whose priority is supporting overburdened installations.

Section 8 discusses some of the assumptions made in modeling beneficiaries and the system of care. Section 9 discusses opportunities for future work including ways to overcome some of the limitations discussed in the preceeding section.

Finally, Section 10 discusses the implications of the experiments and makes policy recommendations.

2 Background

2.1 Post-Traumatic Stress Disorder and Related Conditions in the U.S. Military

2.1.1 Description

Two psychological health diagnoses, Post Traumatic Stress Disorder (PTSD) and Major Depressive Disorder, are conspicuous for their commonality and the rate at which diagnoses have increased since the start of the wars in Iraq and Afghanistan. Multiple deployments have been repeatedly shown to increase the risk of PH conditions for soldiers and marines deployed to Iraq and Afghanistan.

As we would expect from the increase in cumulative number of deployed servicemembers and increases in the average number of deployments per servicemember [Harris et al., 2010, Belasco, 2009], treated PTSD prevalence among screened service members almost tripled between Q1 2005 and Q3 2009 and treated prevalence of Depression over the same period quadrupled [MHS, 2010]. Estimates of untreated prevalence reach into the hundreds of thousands of service members [Tanielian and Jaycox, 2008, Atkinson et al., 2009].

Post Traumatic Stress Disorder (PTSD) Post Traumatic Stress Disorder is an anxiety disorder caused by traumatic or life threatening events. Traumatic experiences including sexual assualt and natural disasters can cause PTSD. In the military, PTSD is most commonly associated with combat. The symptoms of PTSD have been recorded in nearly every war in history, but the official diagnosis of "PTSD" was not added to the Diagnostic and Statistical Manual of Mental Disorders (DSM), the standard reference, until 1980.

Servicemembers exposed to combat are more likely to recieve a PTSD diagnosis than those who are not, and multiple deployments further increase risk [J-MHAT, 2011, Tanielian and Jaycox, 2008, Arthur et al., 2007, Harris et al., 2010].

PTSD is one of many similar anxiety disorder diagnoses including Generalized Anxiety Disorder, Acute Stress Disorder, and Panic Disorder. The U.S. military also recognizes Combat Stress Response, which shares symptoms with PTSD and Acute Stress Reaction, but lasts only a short time [Brusher, None]. While PTSD has been called a "signature wound" of the wars in Iraq and Afghanistan, focusing on PTSD would be misleading. Only 4.2% of mental health encounters in the Military Health System treat explicitly diagnosed PTSD [Harris et al., 2010].

Symptoms of PTSD include re-experiencing the traumatic event(s) that caused its onset, avoiding reminders of the event, and hyper-arousal (being constanly and actively on guard). To be diagnosed, one's symptoms must persist for at least a month [Hablen, None]. By definition, PTSD requires exposure to the threat of injury of death [NIH, None].

Major Depression Disorder (MDD) Major depression is a common mood disorder, and is not necessarily related to combat or trauma.

According to the National Institutes of Mental Health [National Institutes of Mental Health,] symptoms include:

- Persistent sad, anxious or empty feelings
- Feelings of hopelessness or pessimism
- Feelings of guilt, worthlessness or helplessness
- Irritability, restlessness
- Loss of itnerest in activities or hobbies once pleasurable
- Difficulty concentrating, remembering details and making decisions
- Thoughts of suicide, suicide attempts
- Aches or pains that do not ease even with treatment

Some of these symptoms overlap with PTSD and the disorders can occur together.

Rates of depression among servicemembers and veterans are also highly associated with combat exposure.

Other conditions Military beneficiaries seek care for a range of mental health conditions, including those which are not combat related. All patients, whether or not their diagnosis is combat related, make use of the same healthcare system. To accurately assess the demand for care, this analysis accounts for all encounters with mental health providers. The commonality of each diagnoses code in the military population is discussed in depth in [Harris et al., 2010].

2.1.2 Treating and Preventing Psychological Health Conditions

Level of Treatment Psychological health conditions can be treated clinically and non-clinically.

Patients with an official diagnoses can recieve care in an inpatient or outpatient setting. Inpatient care is available to patients with the most severe and complex diagnoses. The patients are admitted to the hospital for the duration of their inpatient treatment and often recieve followup outpatient services after their release. Outpatient clinical care is available for the majority of patients with less severe conditions.

Patients with sub-clinical diagnoses or no diagnoses at all can seek non-clinical care. Non-clinical care can be provided by counselors, chaplains, specially trained Non-Commissioned Officers (NCOs) and others.

This thesis is focused solely on outpatient clinical care.

Direct Care and Purchased Care Servicemembers and their beneficiaries recieve care through their TRICARE benefit.

The military provides some care in its own hospitals and clinics (direct care) and augments this capacity with private civilian providers (purchased care). In the first case, the military pays its own providers. In the latter, the military reimburses civilian providers.

The military prefers to provide psychological health services to its service members through the direct care system. Doing so is less expensive and gives clinicians and commanders more visibility into a soldier's condition. The military takes a keen interest in the psychological condition of its service members because it affects their deployability status and their fitness for various types of duty such as carrying a weapon. Service members have also reported that they prefer to seek care from uniformed providers who understand their expreiences better than civilians [Arthur et al., 2007].

Telehealth Patients can be treated in person or via videoconferencing. Treating a distant patient using telecommunications is known as "telehealth" [ATA, None] and when used for behavioral health care, it is known as "telebehavioral" or "telemental" care. These names are used interchangably in this thesis, but always refer to mental health care.

Most mental health care does not require physical contact and telehealth has been shown to be as effective as in person treatment in several studies. [Hailey et al., 2008], in a review of 72 papers, concludes that "evidence is encouraging". The military is currently expanding its telehealth networks around the world.

A more complete discussion of telehealth policy in the military is presented in Section 2.5.4.

Resources Used for Treatment Patients with a mental health condition may be treated by a primary care provider or a mental health specialist (psychiatrist, psychologist or social worker). This analysis only examines the demands placed on mental health specialists.

Treating PTSD and MDD is time intensive. Outpatient care for PTSD, MDD and other conditions may consist of pharmacotherapy (medication), psychotherapy ("talk therapy"), or both. Talk therapy is obviously time intensive. To monitor efficacy, adjust prescriptions and dosages, and to guard against polypharmacy, pharmacotherapy requires the time and attention of the prescribing provider, such as a psychiatrist, and the patient's other providers, like a primary care physician.

Patients may be treated either in person or by telehealth. The DoD uses telehealth to provide talk therapy and prescription management remotely. Treating patients via telehealth requires at least as much clinical manpower as in-person treatment. Interviewees reported that in some cases it required an especially large amount of administrative overhead because of the inefficiencies in scheduling patients, properly recording the encounters, and because of variation in practices between the patient's installation and the provider's installation.

Pathways to care Several paths to treatment are available to servicemembers and their families. They can choose to seek care from a clinician, a non-clinical couselor (e.g. Military OneSource), from clergy, or from community resources not associated with the military. Any of these caregivers can suggest other classes of caregiver to the patients as the see fit. Because this analysis only accounts for outpatient medical treatment, only pathways to outpatient care are discussed.

There are several different paths a patient can take to get mental health care:

1. Referal from primary care - Physicians may refer a patient to a specialist or a provider embedded in the the primary care clinic[1].
2. Command Directed Mental Health Evaluation - Commanders may order individual servicemembers to be evaluated by a provider.
3. Screening - All servicemembers are periodically screened for health issues including psychological ones. Screening positive for a mental health condition during these assessments can result in a referral. Screenings occur annually and at set intervals before and after deploying.
4. Emergency Room visits - Patients who present with a psychological condition in the ER may be referred for inpatient or outpatient care
5. Chaplain Referral - Chaplains are a first line source of couseling and are part of combat units. Chaplains can refer their soldiers for medical care.
6. Marriage and Family Life Couselor Referral - MFLCs refer patients who need care which is more specialized than they can provide. MFLCs are focused on non-clinical counseling. Seein an MFLC is less stigmatized because they do not keep records of services provided. As a result, patients are often more willing to self-refer to these providers.
7. Self-Referral - Some patients directly seek mental health care from a hospital, clinic, or individual provider.
8. Substance Abuse Referral - Substance abuse co-occurs with a large proportion of mental health issues, especially PTSD [Thomas et al., 2010, Hoge et al., 2004]. Counselors may refer patients for care for an underlying mental health issue in addition to treating them for substance abuse.

2.1.3 Accession Screening

Psychiatric conditions in new recruits are difficult to identify and costly to overlook. Screenings are based on self-reported histories and throughout OIF and OEF, the leading cause of hospitalization during the first two years of service has been psychiatric conditions [Niebuhr et al., 2010].

[1]RESPECT-Mil and the Air Forces' Behavioral Health Optimization in Primary Care (BHOP) are two examples of "primary care integration" programs. In these and others, a psychological health provider is either co-located with a primary care clinic or otherwise made available on a permanant basis.

Improved screenings and changes in screening methodologies could significantly impact the diagnosis rates of new recruits.

Current Screenings All applicants to the military are screened for pre-existing medical issues before joining. Recruits with a history of psychotic, adjustment, behavioral, personality, dissociative, or anxiety disorders, a current mood disorder, or a history of suicidality are rejected except in rare cases where a waiver may be issued [Sackett and Mavor, 2006, Niebuhr et al., 2010].

Screenings are currently used to restrict some duties to those without histories of mental illness or other risk factors. For example, some special operations units will not accept patients with any history of mental illness (even if it is combat related).

Variance Among Service Members The risk of a new service member being diagnosed with a mental health condition varies and can be predicted. When there are more applicants than needed, the military can afford to be selective. In 2009, perhaps because of the recent recession, the military assessed 340,000 applicants compared to the average of 280,000 per year in the previous 5 years. When there are few recruits, the military can not be as selective. Emerging research [Garb, None] tentatively shows that scores on more detailed questionaires can accurately predict mental health diagnosis rates among soldiers within a cohort.

2.1.4 Prevention Efforts

Dozens of efforts across the services are aimed at preventing mental health issues [Weinick, 2011, Meredith, 2011]. Most of these fall into two major categories, resilience training, and operational stress control.

Different services are available to different populations, and service members are not uniformly exposed to each. Some programs are available only to particular services, installations, or military units. Since some of these services are relatively new, seasoned veterans only gained access to them after multiple deployments while newer service members might have been trained before ever deploying.

Resilience Training The services have introduced a host of resilience training efforts (see [Meredith, 2011] for a representative listing). The efficacy of these programs is still being evaluated. These programs are expected to decrease individuals' risk of a future psychological health condition. Such training is not a replacement for clinical care.

Operational Stress Control Each of the services has long operated its own Operational Stress Control (somtimes called Combat and Operational Stress Control, COSC) units. These units provide first line defense against stress reactions in theater. By promoting sleep hygeine and giving servicemembers counseling and/or a few days away from the front line, OSC helps to prevent medical evaluations for mental health issues.

2.1.5 Mental Health and Remotely Piloted Aircraft Operators

Remotely Piloted Aircraft (RPAs), commonly known as drones, are sometimes operated from the battlefield and sometimes operators can be thousands of miles away in the safety of a continental United States military installation, like Creech Air Force Base in Nevada. On battlefield operators likely face similar stresses to those around them. But distance is not enough to isolate off battlefield pilots from all stresses of war.

"Drone Operator" is a fairly new occupation, but drone pilots and sensor operators are in high – and growing – demand [Spiegel, 2008]. Literature on the specific psychological impacts of remote warfare is sparse. Most related literature focuses on human factors or pilot (or sensor operator) selection criteria (e.g. [Chappelle et al., 2010] and its associated references). But, the psychological effects of combat from afar have gained increasing mention in the press.

The Associated Press reports stress on drone operators is causing family and relationship issues [Associated Press, 2008]. Stress isn't necessarily a clinical problem, but Reuters quotes a report by Air Force Research Lab investigators Wayne Chappelle and Kent McDonald which found that 17% of RPA pilots showed signs of "clinical distress" and an even higher proportion reported burnout [Stewart, 2011]. Interviewees at several Army installations reported that issues with family and romantic relationships usually coincided with mental health care seeking.

Having recognized the unique stresses, at least one installation has has enlisted the services of mental health providers to support their air crews[Associated Press, 2008].

Unique Occupational Characteristics Drone operators face a different set of stressors than their traditional pilot bretheren and from other service members.

Because drone operators commute to work, take part in a war, then go home to a civilian life, they don't experience the same camraderie and decompression that service members in theater do [Associated Press, 2008]. Even so, they are asked to reckon with some of the same stressors that are known to cause mental health issues in traditional warfighters. Operators not only watch the impact of their strikes, but sometimes observe their targets for weeks beforehand as part of reconnaisance missions. [Hoge et al., 2004], an authoritative study on the effects of combat in Iraq and Afghanistan, asks a battery of questions on combat experiences related to psychological trauma. A drone operator could concievably see 8 of the 18 situations listed. Those related to personal safety though, would not apply.

Strictly speaking, a drone operator could not get post-traumatic stress disorder from his work. PTSD diagnoses require the initiating experience to involve threat of personal injury or death.

Drone operators are at risk for any of the other countless mental health diagnoses, and ineligibility for an official PTSD diagnoses does not preclude them from other anxiety and mood disorders.

While drone operators don't experience the same risks as traditional combatants, there are several important occupational risk factors to consider. Drone operators often engage in regular operations for years on end [Spiegel, 2008]. Traditional warfighters typically go on a typical deployment which usually lasts less than a year. Long deployments have been repeatedely shown to have a negative impact psychological wellbeing [Arthur et al., 2007, J-MHAT, 2011]. Drone operators can also be especially young for flying combat missions. It's not uncommon for 18-19 year old Airmen to serve as sensor operators [Associated Press, 2008]. Younger service members and those in the enlisted ranks in the general military population have used disproportionately high amounts of mental health care [Harris et al., 2010].

2.2 Access to Care

Access to Care can be defined many different ways. Access, in this context of this analysis, refers the availability of services when and where a patient requires them.

2.2.1 Measuring Access to Care

Access to care in a service system is a function of supply and demand. Simply put, insufficient supply of care results in decreased access for at least some individuals.

Access to care can be measured using many different metrics. [Aday and Andersen, 1974] describe two categories of indicators, utilization and satisfaction.

- Utilization: The level and pattern of the population's actual utilization. In this context, utilization includes information about
    - Who provided the care
    - Where care was provided
    - Why care was provided (e.g. preventative, illness-related, or custodial)
    - How much care was provided. This will vary from person to person and might include
        - Whether or not a person entered care at all
        - The number of encounters a person receieved
        - The continuity of a person's care
- Satisfaction: Patients' impression of the services provided. This can depend on factors such as convenience, perceived quality, coordination, and courtesy.

2.2.2 Decisions Affecting Access to Care

> "Thus far, access has been more of a political than an operational idea." -[Aday and Andersen, 1974]

Access to health care is often the focus of policy making [Aday and Andersen, 1974]. At times, policy making even focuses explicitly on access to specialty services such as mental health (e.g. the Paul Wellstone and Pete Domenici Mental Health Parity and Addiction Equity Act of 2008). Aday and Andersen name four categories health policy focuses on: Financing, Education, Manpower and Organziation. The authors show the impacts of these decisions on both the "Health Delivery System" and on the "Population at Risk".

In Aday and Andersen's framework, two characteristics of the Health Delivery System, resources and organization, might be affected by policy decisions. They consider both the overall volume of resources and the distribution of resources among individuals and geographic areas. Their conception of organization is not limited to structure; it also includes processes such entering the system and referral procedures.

2.2.3 Demand Suppression

When care is not available, too expensive, too inconvenient or too stigmatized, [Keeler et al., 1986, Tanielian and Jaycox, 2008] individuals do not seek care. The difference between the amount of care desired or needed and the amount of care recieved is known as suppressed demand. Often, demand suppression is characterized in relation to price, insurance coverage or demographic factors [Keeler et al., 1986, Wells, 2001].

Within the active duty military, coverage is not a major issue. Servicemembers and their families recieve universal coverage for mental health care through their TRICARE benefit. The military's mental health care coverage compares favorably to private sector plans and to other government plans [Levy et al., 2009]. From the perspective of patient costs, the military provides very good coverage. However, this does not imply that all TRICARE beneficiaries recieve the care they need.

The availability of providers varies widely by location. [Tanielian and Jaycox, 2008], drawing from several sources [Johnson and Sherman, 2007, Arthur et al., 2007] shows that "Service availability is variable and some gaps are reported." The U.S. Department of Health and Human Services' analysis of the US healthcare system confirms that some areas suffer from severe shortages of mental health providers [?]. Sometimes, MTFs are overburdened or do not offer all types of services. For example, some MTFs offer intensive outpatient care, while others do not. In some locations, purchased care providers are hard to find. The DoD's Task Force on Mental Health report [Arthur et al., 2007] tells the story of one direct care provider who called over 100 local network providers listed on the TRICARE website and found that only three were actually accepting new patients.

The utilization of health care varies widely by locale. Often, this reflects the supply of care, not the actual need for care. Using data from the Dartmouth Health Atlas, [Wennberg et al., 2002, Wennberg, 2004] show that supply-sensitive care for chronic illnesses, including specialist visits, are overprovided in some locales with no discernable benefit. In a study of VA mental health care [Rosenheck and Fontana, 2007], the total volume of encounters in the VA system remained nearly constant, while patient load increased. According to Rosenheck and Fontana's analysis, the increase in patient load appears to be associated with a 37% reduction in visits per mental health patient.

Measuring Demand Suppresssion If all patients seeking care recieve a full course of the necessary treatment, utilization and demand are equivalent. Where the system of care is constrained, however, utilization will understate demand. Suppressed demand for real world systems can be estimated using surveys, an experiment (e.g. [Keeler et al., 1986]) or natural experiment (e.g. [Card et al., 2008]). While such methods can provide evidence of suppression under generic contexts or a relatively accurate measure of suppression under specific contexts, measuring suppression in the military is difficult. To date, no compelling data on the suppressed demand in the military is available.

[Levy and Gabay, 2008] reports that the availability of physicians is likely to vary across areas. [Levy et al., 2009] shows that utilization varies drastically by TRICARE option selected and that there is some area to area variation in utilization. Some variation should be expected; demand for and supply of care varies by installation. Demand varies according to many demographic factors, especially deployment history. In addition to the quantity of care available, different installations have different processes for entering the system of care [Scott et al., 2011, Hess and Srinivasan, 2012, Scott and Srinivasan, 2012].

Because of site-to-site variations, suppressed demand cannot be measured by simply comparing per capita utilization across catchment areas.

Accounting for Demand Suppression in Modeling and Analysis The military frequently uses models to determine how much capacity it will need in its direct care facilities. Such capacity planning relies on a credible estimate of future demand. Since the military is a unique population, such estimations usually rely on past utilization data within the military's healthcare system. According to a review in [Harris et al., 2010], many models (ASAM, Red Cell, Physicians Requirement Model) simply ignore suppressed demand. Harris' own model, PHRAMS, explicitly accounts for suppressed demand, but estimates it to be 10% at all sites[2].

[2]The PHRAMS model allows users to set this variable. It's default value is 10% for all the services. The report accompanying PHRAMS does not discuss a methodology for settling on 10%. Determining the actual level of suppressed demand is a responsibility left to the user.

2.3 Personnel Requirements Determination

Site visits, interviews and other analyses suggest that capacity is limited by provider availability. In the Direct Care sector, the capacity of the system is directly proportional to the number of providers and their productivity. In the Purchased Care sector, capacity is constrained by the ability of private providers to accept new patients into their caseload and the willingness of those providers to accept TRICARE's reimbursement rates [Arthur et al., 2007].

Access to care can be improved by ensuring providers are available when patients need them. Insurers, hospitals, and governments use many techniques to predict demand and provide supply accordingly.

2.3.1 The Relationship Between Budgeting and Staffing

Staffing levels in MTFs are heavily dependent on budgeting decions. Therefore, we might wish to analyze the system in terms of the budgetary decisions made, rather than staffing levels acheieved. Indeed, this would be a more salient approach if budget were the only determinant of staffing. However, many other decisions made by policy makers will impact the actual staffing levels in MTFs. Staff must be hired, trained, credentialed and privileged. This can take considerable time. Some positions remain unfilled for months or longer.

Because so many process, policy, and economic factors affect the relationship between budget decisions and actual manpower, this analysis confines itself to discussing resources only in the context of manpower.

2.3.2 General Techniques for Estimating Psychiatric Manpower Needs

[Faulkner and Goldman, 1997] describes five techniques for estimating Psychiatric Manpower Requirements. While these only project the number of psychiatrists needed, they provide a reasonable framework with which to understand mental health manpower needs. Their five methods are summarized below.

1. Based on the current supply of psychiatrists, increase or decrease the requirement based on the perceived adequacy of the system's capacity.

2. Based on the demands of specific service systems (for example, a health maintenance organization's enrollment) and the staffing levels used in that system, extrapolate the need for another given system. This method assumes the adequacy of care provision in the system used as a baseline.

3. Based on data from other countries, extrapolate the need for another given system. The mechanics of the calculations are identical to (2).

4. Based on what the system can afford to spend on psychiatry, determine how many psychiatrists can be hired. This method does not account for patient needs or population size.

5. Based on the projected needs of patients, determine their demand for care and the respective number of psychiatrists required to meet that demand. This method requires more data, which may not be easy to obtain.

Faulkner and Goldman conclude that method five, using a needs-based model, is best for the beneficiaries of the system.

2.3.3 Personnel Requirements Determination for Military Treatment Facilities

Each service operates its own medical system. The Army's Medical Command (MEDCOM), the Navy's Bureau of Medicine (BUMED) and the Air Force Medical Operations Agency (AFMOA), all serve their respective constituency; BUMED also serves the Marine Corps. The medical commands fall under the organization of their respective services.

Official Descriptions The Government Accountabilitiy Office's July 2010 report: "Enhanced Collaboration and Process Improvements Needed for Determining Military Treatment Facility Medical Personnel Requirements" [GAO, 2010] describes the methods each service claims to use. They are summarized below.

Army: The Army's Automated Staffing Assessment Model projects manpower requirements for all medical specialties for each Army MTF. This model is need-based and projects demand for the current enrolled populations of each facility.

Navy: The Navy uses current manning levels as a baseline and adjusts those levels when necessary. Levels may be adjusted to account for emerging needs or for changes in its medical mission.

Air Force: The Air Force uses an ad-hoc system of adjusting manning requirements. MTF commanders or general officers in the Air Force Medical Operations Agency can initiate a process of re-evaluation of needs by facilities. Subject matter experts then produce a new requirement for personnel which is vetted by personnel ranging from MTF commanders to Air Force Chief of Staff.

Our Observations of MTF Staffing Procedures Military Treatment Facilities, the hospitals owned and operated by the military, usually have a set of mental health departments[3] resembling their counterparts in a civilian general hospital. Funding decisions, which are effectively staffing

[3]Typically, an MTF will have a Department of Psychiatry, a Department of Psychology and a Department of Social Work. This is not universal.

decisions, for these departments are made by the MTF's leadership with guidance from behavioral health department leaders.

The budget each MTF recieves is based on its productivity, measured in Relative Value Units (RVUs)[4]. An RVU value is assigned to each procedure. When the MTF provides a procedure, such as outpatient psychotherapy, it is said to have "produced" the corresponding number of RVUs. The MTF is entitled to a budget based on its production of RVUs (each RVU has a dollar value), which it then uses to fund its operations.

Within the MTF, this budget is not automatically distributed to departments based on their production of RVUs. Instead, each department's chief is responsible for negotiating the funding of their department with MTF leadership. Psychological health providers reported that MTF commanders used varying criteria to decide whether or not to approve these requests. Much, but not all, of their justification relies on departmental RVU production. Many departments we visited recieved additional funding because their RVU production was not enough to sustain their department; other departments claimed to produce enough RVUs to fund their own operations and still leave money in the MTF's coffers. In addition to RVU production, some departments may bolster their budget justifcation, citing projected future needs and the department's provision of non-reimbursible services[5].

When asked how staffing decisions are made, department chiefs and other leaders described a system in which they were free to request more staff from their parent organization, the MTF. No interviewee described a system in which staffing models were used to assign staffing quotas to particular departments. This finding corroborates the July 2010 Government Accountability Office (GAO) statement that "The services do not centrally manage their processes for their civilian personnel requirements ... local commanders determine these requirements." The GAO report goes on to say that "...the services may be missing the opportunity to make a strategic determination of how many civilian medical professionals are needed to carry out their expected workloads."

2.4 Models Addressing Similar Questions

This section draws heavily from [Hess et al., 2011].

2.4.1 Civilian Models

A survey of American HMOs [Dial et al., 1995] reports that most HMOs use a target patient to provider ratios, at least for primary care physicians. To further adjust estimates, HMOs reported

[4]Other units of measure are also used, but RVUs are the most relevant to outpatient mental health.

[5]For example, Walter Reed Army Medical Center's (WRAMC) Preventive Medical Psychiatry program visits patients who are not diagnosed as needing psychiatric care. Though the service is non-reimbursible, WRAMC leadership continues to support PMP because it builds relationships with high-risk patients that reduce the stigma of seeking formal care.

using other measures including expected growth, appointment waiting time and geographic coverage. According to the same study, HMOs routinely adjust the ratios based on demographics.

Health Resources and Services Administration The Department of Health and Human Services' Health Resource and Services Administration designates geographical areas of the country which have a shortage of mental health providers [?]. To make this assessment, it evaluates the population-to-core-mental-health-professional ratio and the population-to-psychiatrist ratio of each area. It adjusts the threshold for a shortage based on whether or not the area has unusually high needs. The needs of the population are based on demographics and the prevalence of alcoholism and substance abuse.

Physician Requirements Model The Health Resources Service Administration's Physician Requirements Model projects the need for specialist care in the United States [Greenberg and Cultice, 1997]. The patient-to-provider ratios it uses are adjusted depending on demographics. This model does not account for military-specific issues like deployment severity [Harris et al., 2010].

2.4.2 The History of US Military Psychological Health Care Models

Before the adoption of more sophisticated models, the military allocated its resources either subjectively or based on a set of standards and guidelines. In the 1970's the Army began making an effort to measure the current use of manpower in existing organizations [Cooke, 2003].

MS-3 and Benchmarking The first formal systems used for psychiatric manpower determination relied on measuring productivity and determining desired throughput. Desired throughput divided by individual productivity yeilded a simple estimate of manpower necessary.

In 1983 the Army Health Services Command adopted a set of standards to be applied by individual facilities to determine staffing requirements called the Manpower Staffing Standard System (MS-3) [Cooke, 2003].

In 1992, MS-3 was followed by the Benchmarking System developed by the Army Medical Department (AMEDD). The program was criticized as only focusing on very specific tasks within an MTF and not being thorough enough [Thomsen, 1999]. As a result, the program was short lived and replaced (within the Army) by the more modern Automated Staffing Assessment Model (ASAM).

Automated Staffing Assessment Model ASAM is more sophisticated than previous methodologies because it addresses the question of where to spend the next marginal dollar. The model projects the personnel needs for each type of staff based on patient-to-provider ratios. The model provided a basis by which to prioritize resources within a given military hospital [Thomsen, 1999].

The most important factor in ASAM's staffing projections is the size of the population being treated at an MTF. The model is also advised, to a much lesser extent, by population projections. ASAM uses industry performance data from outside the military to help determine manpower requirements for each medical specialty. ASAM can also be tailored to the characteristics of any given MTF such as patient care hours, staff time spent performing ancillary duties, and provider-to-support technician ratio. With this information the model reports back staffing requirements for each MTF a variety of medical specialties such as physicians, nurses, and dentists [GAO, 2010].

PHRAMS At the urging of the Department of Defense's Task Force on Mental Health, the Psychological Health Risk-Adjusted Model for Staffing, PHRAMS, was developed by the Center for Naval Analysis [Arthur et al., 2007, Harris et al., 2010].

PHRAMS is a need-based model for projecting staffing needs for each MTF across all three medical commands. The model projects the demand for each beneficiary enrolled[6] to an MTF or otherwise attached to a Primary Planning Unit (PPU). The demand for care that each beneficiary will (statistically) require is based on demographics including the deployment history of each service member. The demand is then aggregated for each facility, and the number of providers needed is calculated based on assumptions about productivity of uniformed, civilian and contracted providers and the expected portion of the workforce that each type represents.

PHRAMS outputs a projected requirement for each type of provider (such as psychiatrist or clinical social worker) for each facility for each of the next 5 years. Furthermore, based on current manpower data, PHRAMS reports the "gap" between current staffing levels and projected needs for the upcoming year.

The GAO found that PHRAMS is the only medical manpower requirement tool which covers all services' MTFs [GAO, 2010]. PHRAMS is the only model used by the DOD which calculates the demographic risk factors for each installation's population when determining the demand for care.

Red Cell The Red Cell model, developed before PHRAMS, but in response to the same Task Force on Mental Health recommendation, addresses many of the same concerns.

It is designed specifically for determing MTF staffing requirements and although it does so in a different way, the model incorporates increases in wartime PTSD into its calculations. A more robust comparison of PHRAMS and Red Cell is presented in the PHRAMS report [Harris et al., 2010].

Dynamic Model for Posttraumatic Stress Disorder Among Troops in Operation Iraqi Freedom [Atkinson et al., 2009] describes a model that projects the prevalence of PTSD over the

[6]A beneficiary who lives on base or nearby may (and in some cases must) "enroll" at the local MTF.

coming years. As its name implies, it focuses solely on this diagnosis and the OIF subpopulation of TRICARE beneficiaries.

This model is particularly interesting because it presents the prevalence of PTSD as a time-series and because it projects prevalence based on deployment schedules.

This model does not account for a system of treatment and does not attempt to determine the number of providers needed to serve this population. Instead, the model demonstrates a method for projecting affliction based on combat intensity. Using this approach, the authors are able to compare the affliction rates in different scenarios. The authors use the model to analyze the scenario of a draft[7], but makes no actionable findings on that account. Aimed at the Veterans Administration, this paper proposes two policy actions: to screen 100% of separating service members for PTSD and to provide adequate quantites of evidence-based care.

The model does not directly inform staffing policy, because it shows only the need for care for one condition in one subpopulation. The model also projects the affliction rate across the entire population, not at a per-installation level.

2.4.3 Limitations of Previous Models

This section discusses the limitations of the models above with respect to the research questions described 1.2. The models above were designed to answer different questions in different situations. The state of the art and the data available to designers also varied. The limitations presented here are therefore no reflection of the quality of the above models.

Previous models do not account for variation in demand resulting from the deployment cycle Sophisticated models like PHRAMS and Red Cell account for the effects of deployment as a demographic risk factor. However, all models discussed above, including PHRAMS, calculate staffing needs for each year based on an annualized demand. This approach assumes patients seek care at a uniform rate over the course of a year[8].

Interviews with providers at several sites suggest that demands during post-deployment periods are significantly higher than average. Installations see these surges for at least two reasons

- The population on base swells with returning servicemembers
- Mandatory post-deployment screenings refer many servicemembers in a short period of time

[7]Under the draft modeled by the authors, all servicemembers would serve one tour. Their model projects a net increase the prevalence of PTSD, despite the fact that PTSD risk increases with an increased number of deployments. This is because their analysis shows marginal increase in risk of a PTSD diagnosis is higher for a first deployment than following ones.

[8]In their final report on PHRAMS, CNA explicitly states that their model does not "forecast requirements to meet post-deployment surges."

A more detailed model would examine the changes in demand for care according to where servicemembers are in their deployment cycle. Since large groups of servicemembers (Division, Brigade and Battalion sized elements) deploy and return together, any variation in care seeking that corresponds with the deployment cycle could have a large effect on local demand levels.

Beyond requiring that an adequate number of encounters are available on an annualized basis, previous staffing models do not explicitly model the process of receiving care.

Previous models do not account for the load-balancing properties of telehealth The staffing models above assume isolated populations of patients and providers. Until recently, that assumption was appropriate. Today, telehealth offers patients the ability to connect with distant providers.

When patients and providers can be matched frictionlessly, as might be possible with telehealth, sites with excess supply can support site with excess demand. While many barriers to such frictionless matching exist (discussed in 2.5.4), policy changes may give way to improvements from the status quo. None of the above models can evaluate the impact of different telehealth network topologies on access to care.

Previous models do not account for site-to-site variations in the system of care Because sites use different architectures in their care delivery system, not all sites will exibit the same behaviors. Some sites treat patients more efficiently (for example, by reducing triage). Some sites have different screening protocols and outreach efforts and therefore diagnose patients at different rates. The purchased care network at each site also varies; metropolitan areas can usually offer a wider variety of services in a greater quantity.

Explicitly modeling such variation is nearly impossible. There are dozens of military hospitals around the world, and even an accurate snapshot would quickly be out of date. Limited data on outcomes and efficiency make simple analyses impractical, so any model explicitly representing site architectures would likely rely on studies of each individual site.

Previous models do not account for the effects of prevention and outreach efforts To stem the tide of new diagnoses, several programs have been instituted at the DoD and service levels. Subclinical treatment programs like Combat and Operational Stress Control and MFLCs may reduce the need for clinical care in a population. Training efforts, like the Marine Corps' OSCAR and the Army's Comprehensive Soldier Fitness may reduce the incidence of PTSD by preparing service members for the stresses of combat. Previous models don't explicitly model the effects of these programs. Many implicitly account for them by extrapolating need based on past trends in past demand.

If these programs are as effective as they're hoped to be, a robust model of the system would account for their impacts on the need for care.

Previous models do not enable a robust evaluation of installation specific purchased care networks The underlying assumption in the design of the above models is either that the purchased care network will absorb whatever the direct care network does not (e.g. ASAM) or that the purchased care network will absorb a set portion of encounters (e.g. PHRAMS). A robust model would account for the *limited* ability of the network to absorb excess demand.

2.5 Military Mental Health Policy

Countless individual policies impact mental health in the military. The ones presented below are particularly important to the research questions addressed in this thesis.

2.5.1 Access-to-Care Standards

MHS's "Guide to Access Success" [MHS, 2008] defines Access to Care (ATC) as ensuring that "...beneficiaries get to the right provider at the right time at the right place."

The guide further explains that:

> "The goal of access management is to implement and sustain a systematic, proactive, and responsive MTF access plan for all clinics and services that meets or exceeds the ATC standards stated in 32 Code of Federal Regulations (CFR) 199.17."
>
> In this analysis, the relevant section of 32 CFR 199.17 concerns allowable appointment waiting times.

Allowable Appointment Waiting Time MHS sets the following allowable waiting times based on the urgency of care [?].

Emergent Emergency services shall be available and accessible to handle emergencies within the service area 24 hours a day, seven days a week.

Urgent The wait time will generally not exceed 24 hours.

Routine The wait time shall not exceed one week.

Specialty The wait time shall not exceed four weeks.

Well-Patient The wait time shall not exceed four weeks.

Guidance from MHS clarifies these requirements [MHS, 2008, Woodson, 2011] for the context of mental health. First appointments with MH providers fall under the requirements of routine care, while follow-up appointments fall under the specialty care requirements unless the provider determines more urgent care is required. [Woodson, 2011] further states that patients may choose to recieve their initial routine visit from either their Primary Care Manager (PCM), an integrated mental health provider within their primary care clinic if available, or directly from a behavioral health care provider[9].

MHS's new guidance should help improve access, but it does not always work as intended. At least one site visited by the PTSI team triages new patients immediately, but then requires most to wait 28 days to begin care with a new provider. For non-urgent patients, this does not speed up entrance (it actually delays it because patients may wait a few days for triage). It hurts continuity of care because patients must retell their story to the new provider from whom they will get care [Scott and Srinivasan, 2012].

Continuity and Access to Care Continuity between patient and provider is especially important in mental health care. Providers build rapport with patients and servicemembers have expressed frustration with needing to "re-tell their story" to several providers. MHS's "Guide to Access Success" [MHS, 2008] states that

> "A patient may waive ATC standards and request appointments outside of ATC Standards for convenience reasons *or to maintain continuity with their provider*, even though an appointment was offered within ATC standards." (emphasis added).

Therefore, so long as an appointment is available with any provider within the appropriate amount of time, the request is considered to have met the access to care standards.

Utilizing the Purchased Care Network Servicemembers must be preauthorized to recieve non-emergent mental health care in the network. All other beneficiaries are automatically authorized to recieve up to eight encounters, but require preauthorization beyond that point [MHS, 2008].

2.5.2 Priorities

Federal regulations explicitly prioritize service members in the direct care system [?]. MTFs can offer care to other types of beneficiaries if they are not completely occupied with demand from active duty service members (ADSMs). This regulation does not explicitly mention telehealth, but has been applied to its care provision as well. One interviewee reported that even though his TH provider cell had made child psychologists available (a critical shortage profession at many sites) that their time was used by distant sites to treat service members. Because of this precedent, this analysis takes this priortization scheme as a given.

[9] Previous guidance [MHS, 2008] specified that patients' first visit could be with any of the above, but did not specify that it was the patient's choice.

Table 1: Deployment Related Screenings Required By Law

Title	Timing
Pre-Deployment Health Assessment (Pre-DHA)	Within 60 Days prior to deployment
Post Deployment Health Assessment (PDHA)	Within 30 days prior to or after redeployment.
Post Deployment Health Re-Assessment (PDHRA)	Between 90 and 180 days after redeployment
Periodic Health Assessment (PHA)	Annually. These assessments meet the [?] requirement for an assessment 7-12 months and 16-24 months from redeployment.

2.5.3 Periodic Screenings

Since 2009, federal law has required "person-to-person" screenings of service members deploying in support of contingency operations (see Figure 1, based on [?, ?]). Although some service members were screened prior to the 2009 law, screening was not uniformly instituted across the services. Mandatory screening is designed to identify service members in need of treatment for PTSD and other conditions.

Screenings have the potential to create a short but pronounced influx of patients because they occur regularly and because service members often deploy, redeploy, and go through screening in large groups. Estimates of the prevalence of mental health conditions in service members implies that hundreds of soldiers could be diagnosed and referred to treatment within a few days[10].

Because they generate somewhat predictable increases in demand for care, timing these screenings properly offers the DOD the potential to manage their demand patterns. Installations may choose to spread screenings out or to plan them for times when they would otherwise see a dip in demand (such as after the deployment of a brigade or division sized unit).

2.5.4 Telebehavioral Health

Telebehavioral health has attracted increased attention within the military during the wars in Iraq and Afghanistan. The military and others have used telebehavioral health for decades. Lately, widespread broadband availability and increasing need for mental health services have made telebehavioral health especially attractive.

Telebehavioral health can improve continuity of care by keeping providers connected to patients who move to another installation or who are deployed. So long as the patient remains a TRICARE beneficiary, they may continue to see military providers. Without telehealth, patients who relocate must find a new provider and rebuilding rapport (or not seek out a new provider at all). With telehealth, patients can continue to see their original provider.

[10] e.g. 5,000 soldier brigade combat team * 10% diagnosed prevalence (conservative) = 500 diagnoses

Because a provider using a video conference link can (technologically speaking) treat a patient anywhere there's an internet or phone connection, telebehavioral health offers the opportunity for distant providers to pitch in and help installations where demand for care outpaces supply. When installations experience periodic surges, TH allows distant providers to help with screenings, like PDHAs, administrative duties, such as Medical Evaluation Boards (MEBs) or can take new patients into their own caseload. Some installations, especially those in rural areas, find it difficult to hire enough providers and have a persistent shortage of providers. By properly augmenting local care capacity via TH, excessive wait times and demand suppression can be minimized.

Under a new law[11] [?], federal healthcare providers may serve patients by telehealth (including telebehavioral health) no matter where patients or providers are located, so long as the service is within the scope of authorized federal duties. Until passage of the 2012 NDAA, service members and military providers could only use telebehavioral links from one federal facility to another. That meant that providers and patients generally had to be located at Military Treatment Facilities (MTFs). This limited the access for rural service members and prevented the military from hiring providers who did not wish to relocate to an MTF.

Patient views of telebehavioral health are mixed. Some patients prefer the modality while others abhor it. Some appreciate the increased anonymity it offers. TH can reduce stigma for these patients because they know that their provider does not personally know others in their community. Some find it easier to share personal feelings with a distant provider. Others find the modality impersonal and insulting, and strongly prefer an in-person provider. The military has not mandated that beneficiaries use TH. To date, service members have been provided the option, if available.

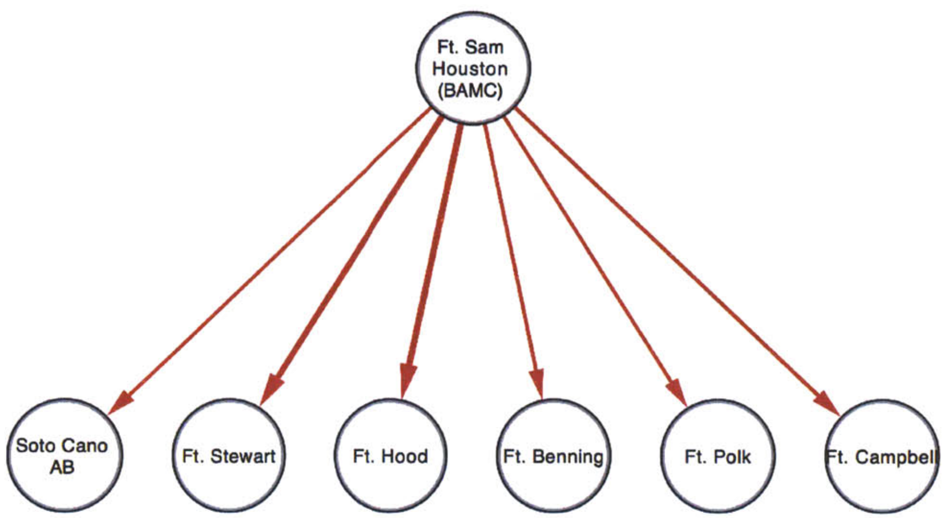
Telehealth Network Topologies Today, telehealth networks operate on several different network topologies. Connections are most often between a pair of sites (for example, Joint Base Lewis-McChord and Ft. Carson staff augment one another during surges, Figure 2c) or operate on a Hub-and-Spoke model. The Walter Reed National Military Medical Center, for example, provides TBH care to patients at sites spread throughout their region of the Army's Medical Command (MEDCOM), but generally does not utilize providers at remote sites to augment its own staff (Figure 1 shows the Southern Region's TH cell at Ft. Sam Houston and recipients of their care).

Administrative and technological barriers have prevented providers and patients from connecting between any arbitrary pair of bases. Until the December 2011 changes, providers were required to procure credentials and privileges from each MTF they delivered care to. This process is time consuming, and duplicative of the credentialling and privileging processes the providers had completed at their home facilities. Patients and providers still have difficulty coordinating appointments across sites since each site runs its own scheduling system. Because of these barriers, operating even a small TH network has required a great deal of administrative overhead.

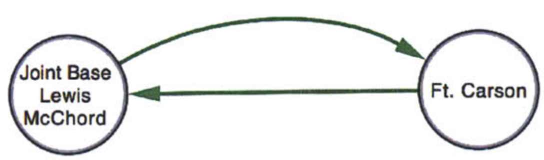
[11] Formerly introduced as HR 1832: Servicemembers' Telemedicine and E-Health Portability Act of 2011, or the "STEP Act"

Figure 1: Examples of Current Telebehavioral Network Topologies

(a) Legend



(b) Hub and Spoke Model



(c) Pairwise Link Model

Under the newly introduced law, credentials and privileges from one installation will allow any provider to serve patients at any facility and even in their own homes. This offers the potential for new TH topologies. The Army is considering consolodating its TH efforts. To that effect, its medical command has issued common guidance on TH operation procedures [Army, 2012].

Telehealth Care Providers Some providers are dedicated to telehealth and others split time between telehealth and other duties.

Several regional medical commands operate a telehealth provider cell (TH cell), which provides care to other sites in their region. Providers in this cell support other installations and usually provide a set number of hours per week to each site to use as they see fit. Usually, sites utilize this spare capacity to perform adminstrative tasks, such as medical board evaluations where providers assess the psychological disability level of service members who are separating from the military.

Other times, providers will take time from conventional care to provide TH. This can be part of a surge support effort, or it can be done to allow a patient who has moved to a new installation to keep contact with his provider. In at least one region, providers at a research center split time between research duties and telehealth care provision.

3 Scope of Analysis

This thesis uses a two piece model to answer the questions posed in Section 1.2. The first piece of the model is used to estimate demand for mental health care. The second is a simulation used to evaluate the system of care's performance under different scenarios.

This simple model is more trustworthy because it requires fewer tenuous assumptions [Utley and Worthington, 2012]. Credibility is especially important to this model, given its focus on policy making. A more complex model could certainly have answered the research questions above and may be more realistic.

Consideration of Suppressed Demand The most important decision in scoping the model was to extrapolate future diagnosis rates from past ones. Because rates are extrapolated and because our data set has no records of "would be" encounters, our analysis will inevitably understate the diagnosis rate of a system with unlimited capacity.

Extrapolation will yeild conservative estimates not only of demand but also of demand fluctuation. Demand fluctuations would be understated because large cohorts (e.g. approximately 4,000 soldier brigade combat teams) move through the system together. When that cohort would see the highest diagnosis rate, the system is busiest, and therefore less likely to accomodate them. The peak utilization from this cohort of soldiers would therefore be less pronounced than it would have been in an unconstrained situation.

Rigorously quantifying and tracking demand suppression is a promising area for future research. See Section 9.

Accounting for Encounters instead of RVUs Although modeling care utilization in RVUs per patient holds some appeal, this thesis projects demand in visits with a mental health provider per patient, which are referred to as "encounters". Encounters for outpatient mental health care are usually about one hour, and other models have established methodologies for determining provider productivity and demand for care in terms of encounters [Harris et al., 2010].

Clinical Care Provision Substitutibility between clinical and non-clinical care does not need to be explicitly modeled. This model only addresses clinical care. Because this analysis elucidates demand *patterns* instead of magnitudes and because one can reasonably expect demand for non-clinical services to follow the same pattern as demand for clinical care.

Formally Recorded Encounters Only encounter types for which we have a robust record are incorporated into this analysis. Because the M2 database does not contain records of informal clinical care, such as walkabout counseling, we do not include that workload in this analysis.

Not all mental health care is conducted under the auspices of the MTF. Stand-alone programs, which are managerially and financially independent from the MTF, provide a range of services. There are several different ways these programs can be funded and staffed. These stand-alone programs seldom provide clinical care. When they do, it is often in small quantities as part of a research program. This thesis does not explicitly account for their contribution to clinical care provision.

Risk Adjustment This thesis uses a novel method of risk adjustment. Risk (expressed as the expected number of diagnoses per day for a given individual) is calculated based only on branch of service, position in deployment cycle and beneficiary category. The expected diagnosis rate for each installation is then calculated as a daily granularity.

This represents an improvement over previous methodologies which only consider demand per annum.

Static Staffing Levels In reality, staffing levels at installations will fluctuation based on hiring, attrition, deployments of providers, and temporary duty assignments. Furthermore, it may take new hires up to a year to become credentialed, priveliged and trained for a particular location. Delays of several months for credentialling and privileging at distant sites for TH providers are not uncommon.

As this process is difficult to credibly model (given the compartive difficulty of hiring in different markets), this model wishes away this very real systemic behavior. As a result, capacity for care provision in the model is more constant than it would be in real life.

In Garrison The care system in theater is very different from the one in garrison, and the assumptions underlying this analysis would not be appropriate for modeling in theater care. This thesis' analysis of care utilization rates assumes that all encounters in the M2 happen in garrison (at the beneficiaries home station) and the simulation only evaluates access to care in garrison.

Provider Types This thesis considers all providers to be interchangable. It does not explicitly model provider mix, and it assumes that all patients who get care in the simulation get it from a provider commensurate with their needs (a clinical social worker, a psychologist, a psychiatrist or a more specialized provider like a child psychiatrist). In part, this is because there is some substitutability between provider types. It is also because the version of the M2 database used here does not contain provider information or information about patient acuity.

4 Deployment Related Care Seeking Behaviors

This section documents historical patterns of care seeking for active duty service members and their families. This picture of care seeking is further examined to show the pattern with which new patients arrive to the mental health system of care. Utilization rates provide a simple overview of care seeking and arrivals to the system are useful for simulating care provision.

Utilization and arrivals to the system of care are extrapolated from past data. This poses two limitations. First, because we do not have evidence that sufficient care has been provided in the past, the results are not suitable for estimating the overall magnitude of demand. Therefore, these projections are not suitable for projecting staffing needs. Second, because patients who recieved care in the past did not get care immediately upon demand, their utilization is a lagging function of their actual demand. This lag may vary, and is likely to be especially long during the periods of highest demand.

4.1 Method for Correlating Care Reciept with Deployments

Interviewees and literature sources describe post deployment surges in demand. This is not surprising, since the population served by the local healthcare system can increase by 4,000 to 20,000 persons upon the return of combat brigades or divisions. Literature also shows that the prevalence of mental health issues in active duty service members [Hoge et al., 2004, J-MHAT, 2011, Arthur et al., 2007, Tanielian and Jaycox, 2008], their dependents [Mansfield, 2010], veterans [Seal et al., 2009], and members of the national guard and reserves [Thomas et al., 2010, Milliken et al., 2007] increases as service members are exposed to combat. Some of these sources suggest or explicitly show trends in care utilization over the course of multiple deployments.

This analysis expands on the care utilization behaviors demonstrated in these sources and quantifies historical care utilization and historical arrival rates across the deployment cycle and across different classes of beneficiaries within each of the services. Because surges are only temporary (a few weeks), we decided to analyze the post-deployment period at a daily granularity instead of as a whole like in other models. This granularity shows trends which other research had not yet quantified. These trends in were the combined result of patient decisions and the system of care itself.

4.1.1 Implementation

This analysis relied on the Army's M2 data[12]. A computer algorithm (see Appendix B) analyzed the MHS data and generated the demand (expressed in encounters per day or arrivals per day) for each possible beneficiary status which is described by a unique combination of $B =$

[12] Our data set, which covers fiscal years 2003 to 2009, includes demographic and deployment information, catchment area, and encounter records for approximately 5M beneficiaries in each year. This data set is described in greater detail in Appendix A.

$BeneficiaryCategory$, $P = PositionInDeploymentCycle$, and $S = BranchOfService$. The possible values of B, P, and S are listed in Figure 2.

This allowed us to look at service members' care seeking rates before deploying, during their deployment, the first day after their return, the second day after their return and so on. To determine the expected number of encounters per day for a given beneficiary status, the algorithm counts the number of encounters recieved by beneficiaries in a given status and divides that by the number days all beneficiaries spent in that status. Arrivals to the care system are calculated similarly, but only encounters which do not follow within 8 weeks of another are counted (we count patients who have been out of care for 8 weeks as new arrivals).

Example 1: If 10,000 Marines went on their first deployment, and 10 of them recieved an encounter on their fifteenth day back, we would divide 10 encounters utilized by persons in the status $B = ACT$, $P = 1st - 15$", and $S = Marines$ by the 10,000 man days spent in the same status. We would find that $.0010\frac{encounters}{person*day}$ are consumed by Marines in that status.

Example 2: Similary, we could evaluate care recieved by Army dependents during the families' service members' deployments (Represented by the status $B = DA$, $P = during$, and $S = Army$. If 15,000 dependents were enrolled during a one year deployment of their service members, a total of $15,000persons * 365days = 5,475,000PersonDays$, and over the course of that deployment, they combined to utilize 3,750 encounters, we would find that Army dependents seek care at a rate of $\frac{3{,}750encounters}{5.475MDays} = .0006\frac{encounters}{person*day} = .25\frac{encounters}{person*year}$ while their sponsor is deployed.

4.2 Care Utilization Rates

As hypothesized, beneficiaries in each combination of B, P, and S use care at a unique rate. The rates for Active Duty service members and their beneficiaries are summarized below. Analysis is limited to these two beneficiary categories because Active Duty Service Members (ADSMs) and their dependents (DAs) make up 66% of all beneficiaries and an even higher portion of encounters. These families also account for the lions share of deployments, the event of interest for these analyses.

Because services differ, direct comparison of care utilization rates (Asking: "Who recieves more encounters, a soldier or a Marine?") can be misleading. Instead, this section analyzes changes in mental health care utilization across the deployment cycle within each service.

Services differ. Their service members are deployed for different lengths of time. Marines typically deploy for six to seven months at a time, while Soldiers typically deploy for 12 (this was raised to 15 in 2007 and subsequently lowered). Their experiences during deployments are also different. A pilot will see different types of combat than an infrantryman. They deliver care and screen their service members in different ways and they may use different practices for recording these encounters. At home, Soldiers recieve most care though official channels at the installations MTF

Figure 2: List of Beneficiary Statuses

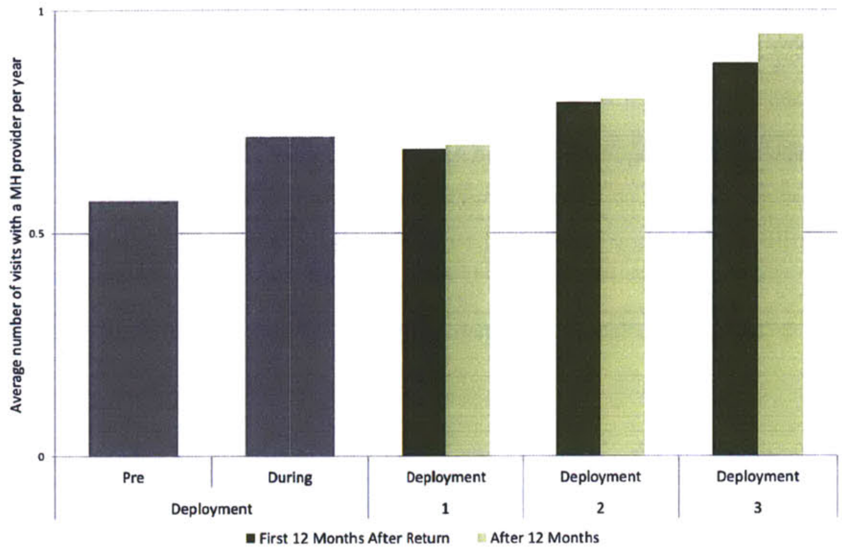
- Branch of Service
  - Army
  - Navy (incl. Marines)
  - Air Force
- Beneficiary Category:
  - ACT: Active Duty
  - DA: Dependent of Active Duty
  - GRD: Active Guard/Reserve
  - DGR: Dependent of Active/Guard Reserve
  - IGR: Inactive Guard/Reserve
  - IDG: Dependent of Inactive Guard/Reserve
  - RET: Retired
  - DR: Dependent of Retired
  - DS: Dependent (Survivor)
  - OTH: Other
- Position in Deployment Cycle
  - Pre: Have never deployed
  - Post: Have deployed but have since changed beneficiary category (e.g. retired)
  - During: Currently deployed
  - Days since First Deployment
  - Days since Second Deployment
  - Days since Third (or later) Deployment

or clinics. Marines usually have providers embedded in their unit (usually from their parent service, the Navy). These differences may have a significant impact in how and when care and screenings take place and get recorded.

4.2.1 Families

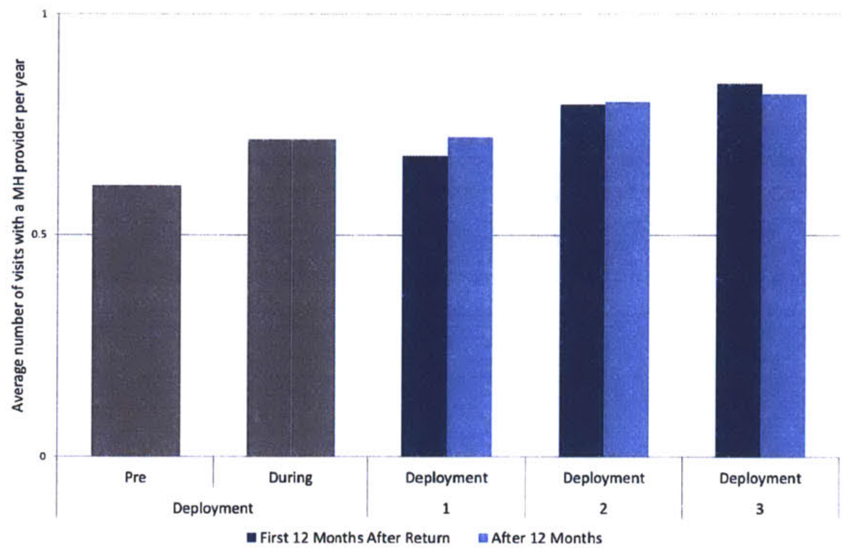
Dependents of ADSMs in every service saw an increase in care provided to family members during deployments as compared to before deployments. In the year after their service members returned home, these rates climb even higher compared to the same baseline. Each marginal deployment was followed by an increase in the rate at which family members utilized care. Family care seeking rates are not presented at a daily granularity because they are steady or gradually increasing over time.

Figure 3: Care Utilization by Dependents of Active Duty Soldiers



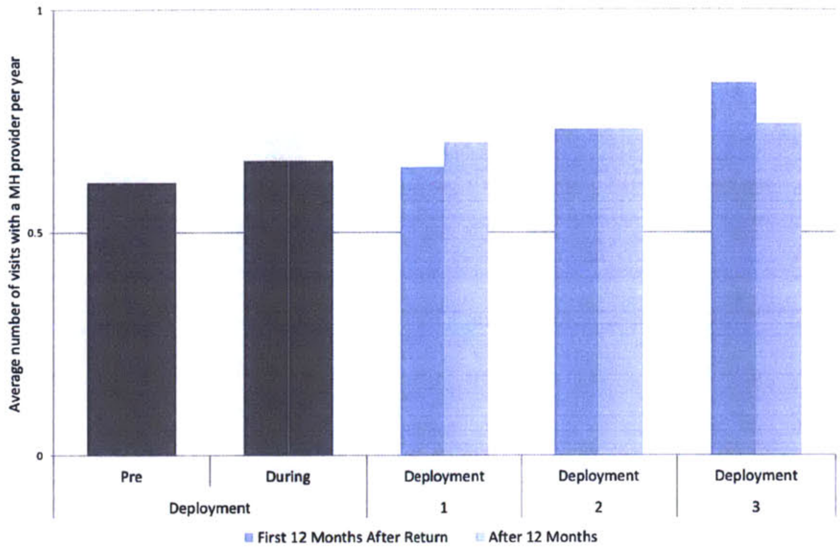
Army Families seek more care during their deployements (+25%) than they do before their service member has ever deployed. The seek care at roughly the same rate in the time following their service member's return from their first deployment. With each subsequent deployment, Army family care utilization rises. After the third deployment, families utilized care at a rate 50% higher than their pre-deployment rate.

Figure 4: Care Utilization by Dependents of Active Duty Sailors



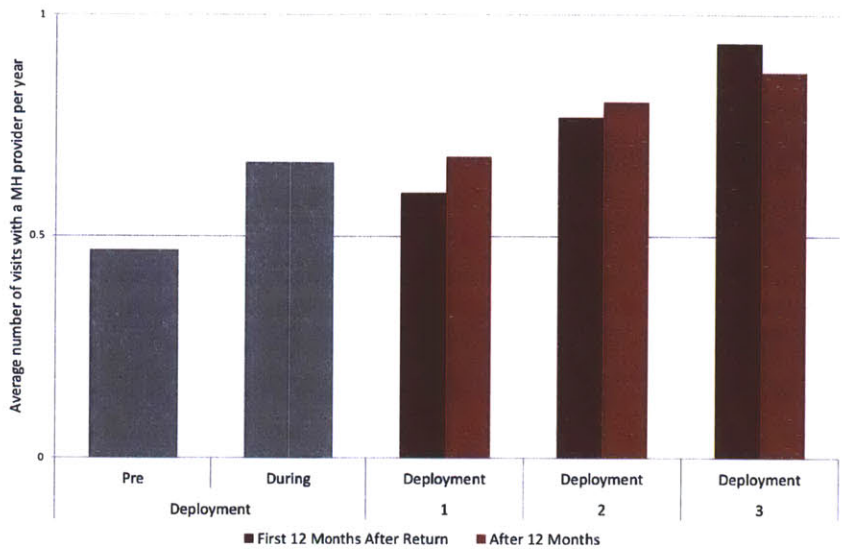
Navy Families also seek care at higher rates during deployment than before (18% more). Navy families recieved slighly higher volumes of care in the periods of time following their service member's return.

Figure 5: Care Utilization by Dependents of Active Duty Airmen



Air Force Families exhibited much less pronounced variation during and following deployments. Care utilization went up by 8% during deployments, and care utilization following deployments goes up about 20% (+37% in the year following the third deployment).

Figure 6: Care Utilization by Dependents of Active Duty Marines



Marine Corps Families show significant increases in care utilization during deployments (+42%) and in times following deployments. In the first 12 months after the first, second and third deployments, Marine Corps dependents used 27%, 64% and 99% more care respectively.

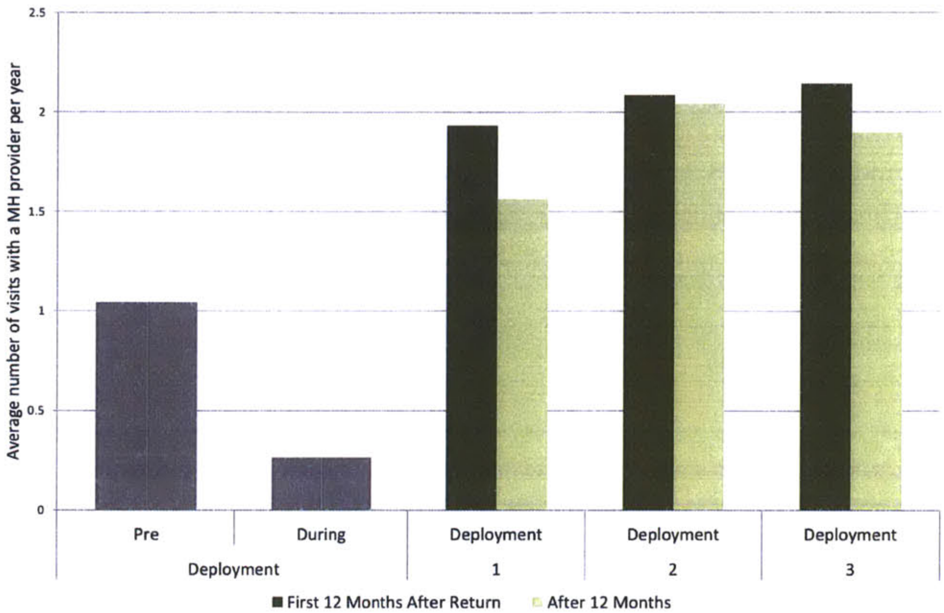
4.2.2 Active Duty Service Members

Army Figure 7a shows that soldiers utilized care at drastically higher rates in the year following their return from theater than they did before ever deploying (1st Deployment: +85%, 2nd Deployment: +100%, 3rd Deployment: +105%). Additional deployments were followed by slight increases in care utilization, but the largest change is between before ever deploying and returning home the first time. During deployments, only a small number of official encounters were recorded. Our data set did not indicate whether these were in theater or attributed to the home station (for example, during brief leave).

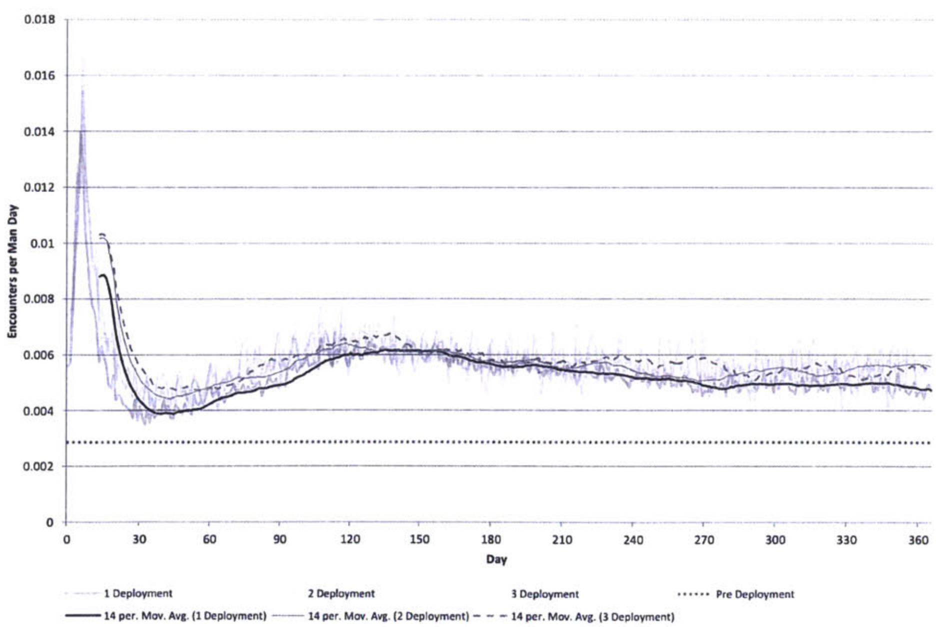
Examining the utilization across the first 12 months after each deployment (Figure 7b), we can see a distinct pattern. An initial spike in utilization upon return is followed by a dip in receipt of care. Based on field work, we suspect that this dip is the result of block leave (the trough) during which most Soldiers go home to see family. After this dip, care utilization steadily rises, peaking between 120 and 150 days after return at more than double the pre-deployment care seeking rate. After this peak, utilization trails off for the remainder of the year. Utilization patterns after the first, second and third deployments follow the same trends. At all times following deployments, soldiers utilized care at significantly higher rates than they did before deploying.

Figure 7: Care Utilization by Active Duty Soldiers

(a) Over multiple deployments



(b) In the 12 months after returning from a deployment

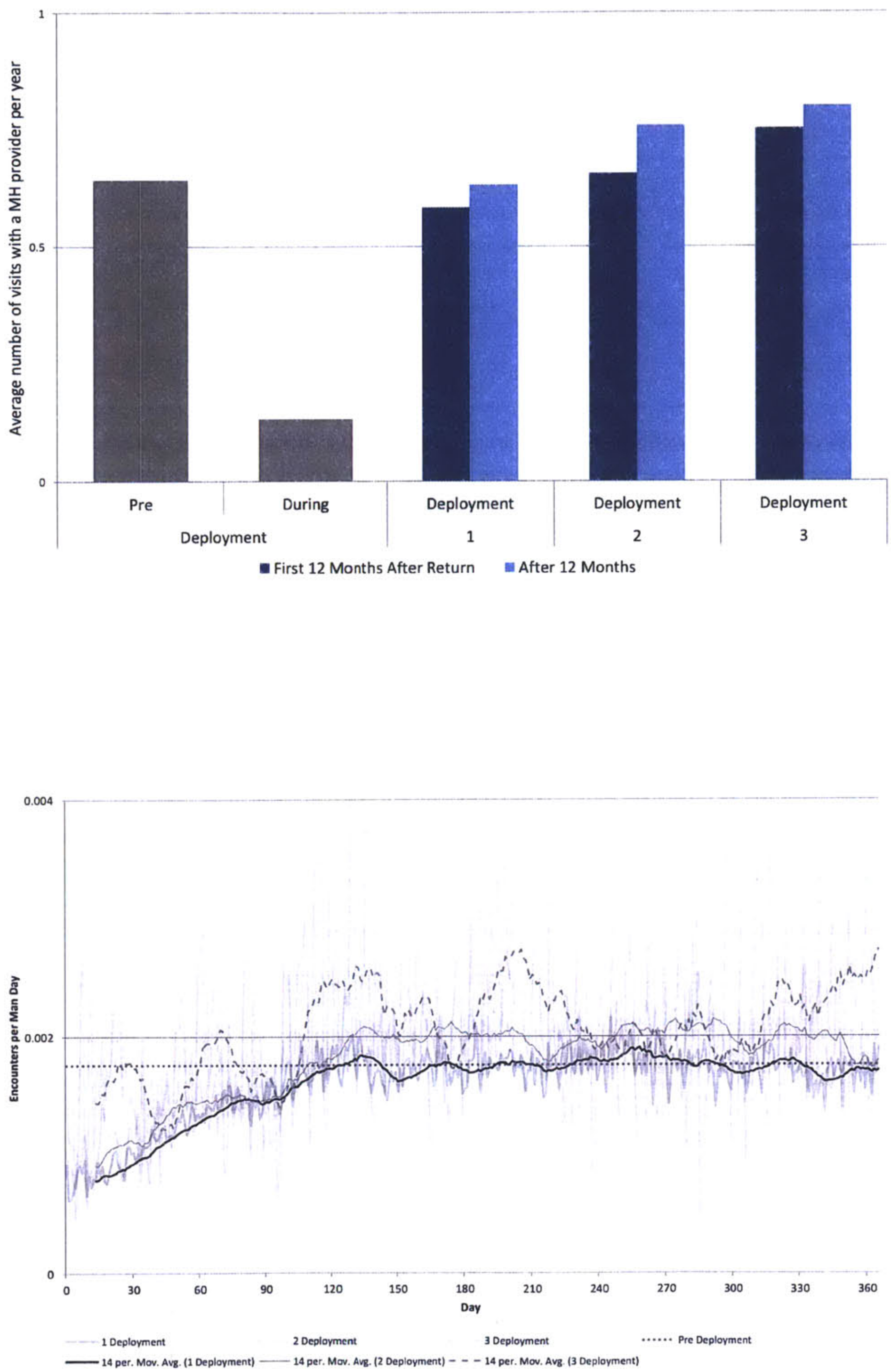


Navy Sailors used less care while on deployment than before or after. In the first 12 months after each deployment, sailors used less care than they did in the time after the window (see Figure 8a). Looking at the daily granularity breakdown of post deployment demand in Figure 9b, we can see that this is probably because sailors use very little care in the first four months after deployment. Once they've been home for four months, sailors utilize care at utilize care at close to predeployment rates.

Rates following the third deployment fluctuate widely, but this may because of a small sample size. While there were more than 213,000 sailors in our data set who had a first deployment, only about 10,000 went on a third deployment. Obviously, only a subset of these used the Navy's mental health care system.

Figure 8: Care Utilization by Active Duty Sailors

(a) Over multiple deployments

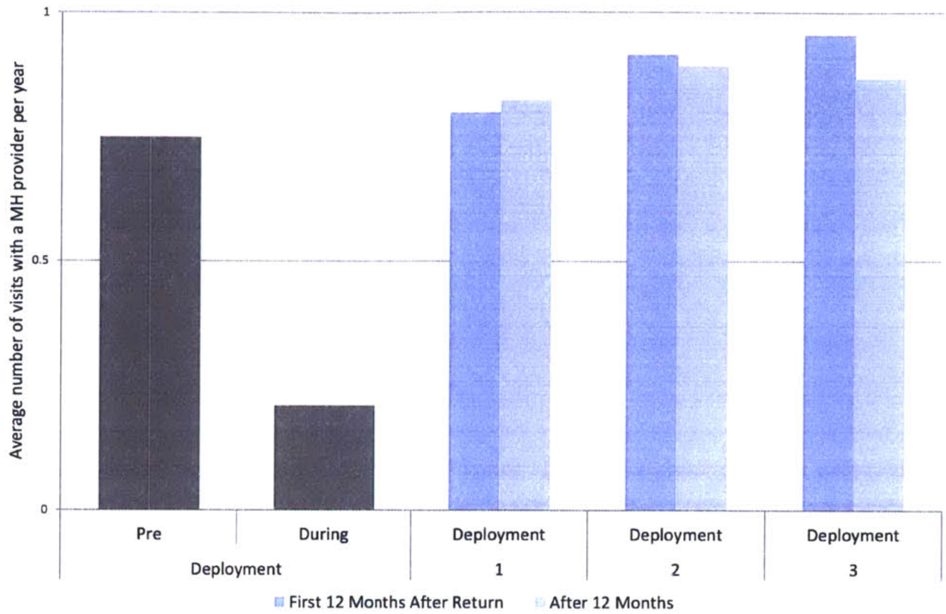


(b) In the 12 months after returning from a deployment

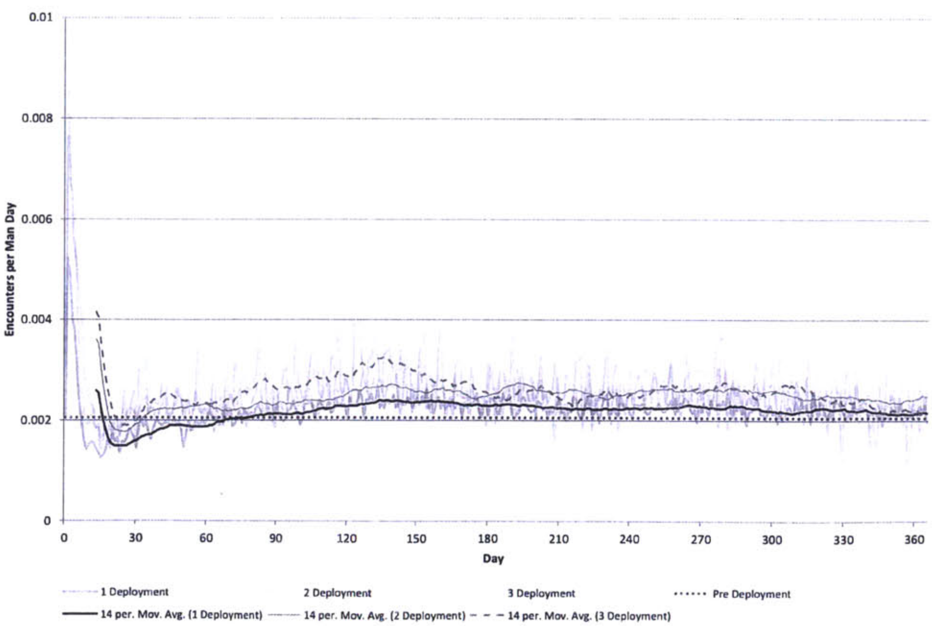
Air Force Airmen used more care after deployments than they did before, but care utilization did not increase as dramatically as it did for soldiers (Figure 9a). Like other service members, they used very little care while deployed. Looking at their utilization in the 12 months following returns from theater (Figure 9b), we can see similiar but less prounounced versions of the trends soldiers displayed. An initial peak is followed by a dip and a slight rise and peak at about 4 to 5 months after return.

Figure 9: Care Utilization by Active Duty Airmen

(a) Over multiple deployments



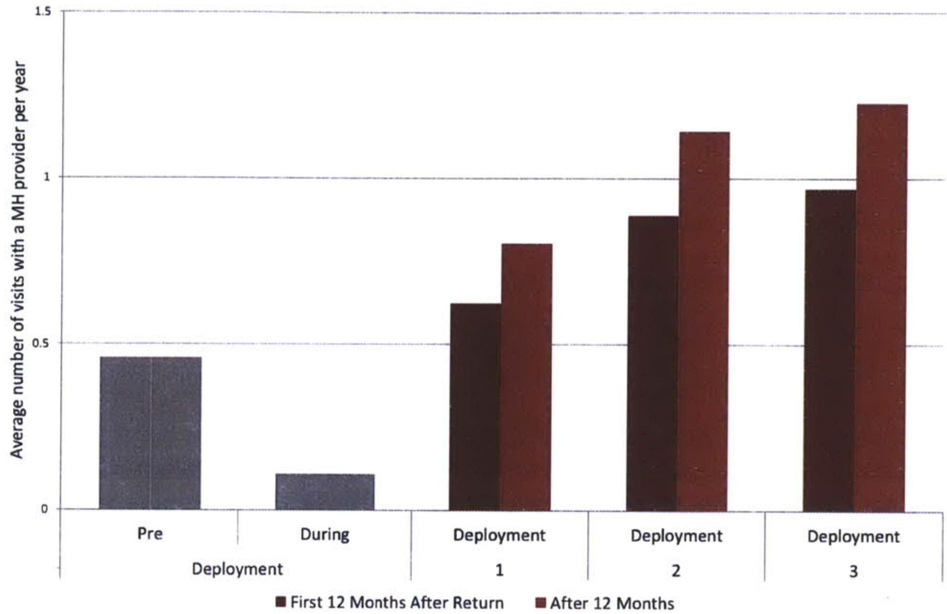
(b) In the 12 months after returning from a deployment

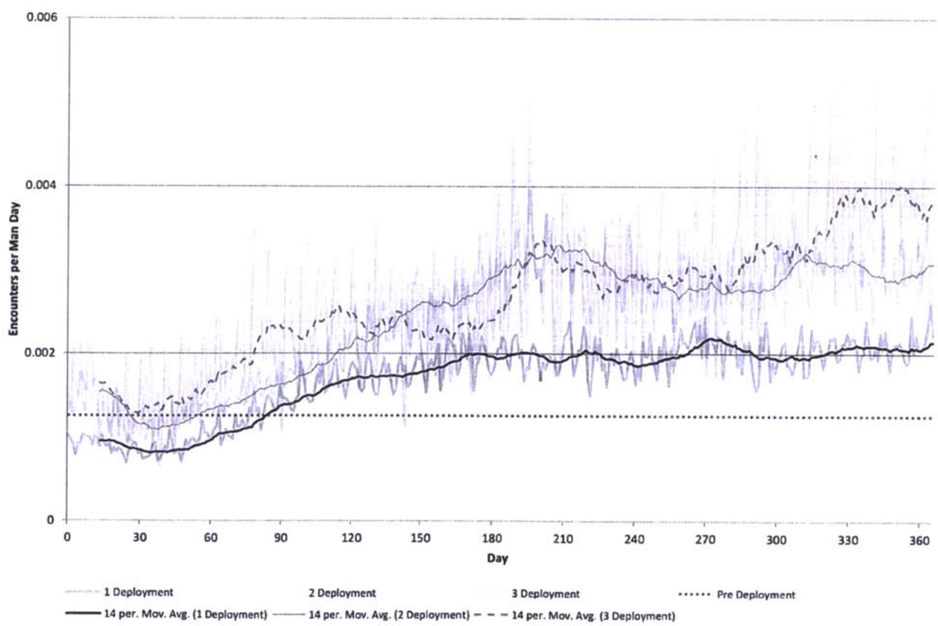


Marine Corps Marines, like soldiers, used drastically more care after deploying than they did beforeband (1st Deployment: +36%, 2nd Deployment: +94%, 3rd Deployment: +111%, shown in Figure 10a. Each marginal deployment was followed by significant increases in care utilization. Interestingly and unlike soldiers, they used care at a lower rate in their first 12 months home than afterwards. Figure 11b shows care utilization upon return was initially low, and rose over the first six or more months home. After spending six months home, Marines used care at rates far higher than they did before deployment.

Figure 10: Care Utilization by Active Duty Marines

(a) Over multiple deployments





(b) In the 12 months after returning from a deployment

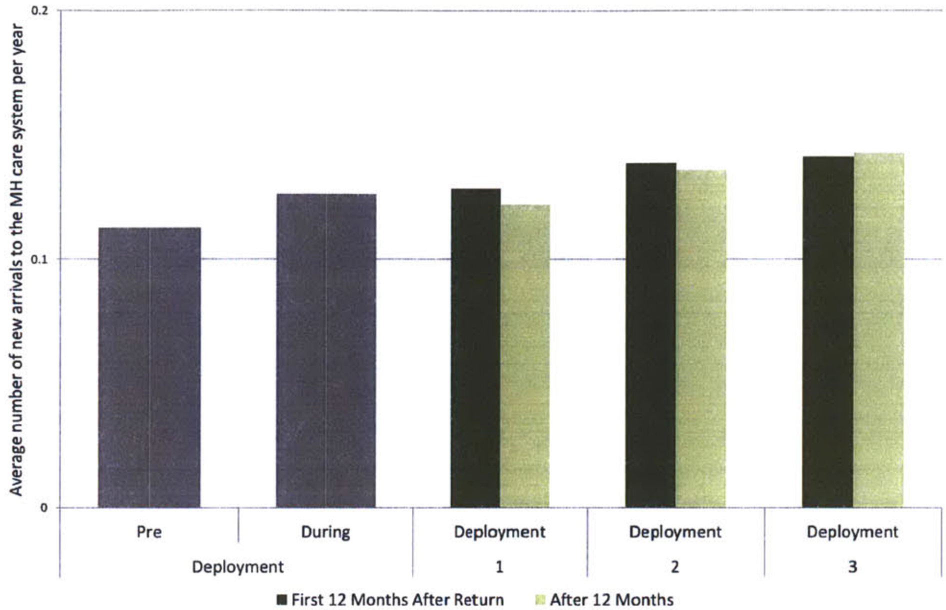
4.3 Arrivals to the System of Care

The care utilization rates in 4.2 tell only part of the story. They tell us how many total visits patients had with mental health providers, but not when each of those patients first arrived to the system of care. This section describes the rate at which new patients arrive to the treatment system. To do this, each patients' encounters are grouped into "episodes" of care. Episodes here are defined as any sequence of encounters with less than 8 weeks between any two. Because most episodes consist of more than one encounter, looking at the start of episodes can elucidate patterns in care seeking behavior that would be obscured by data which showed all care usage.

4.3.1 Families

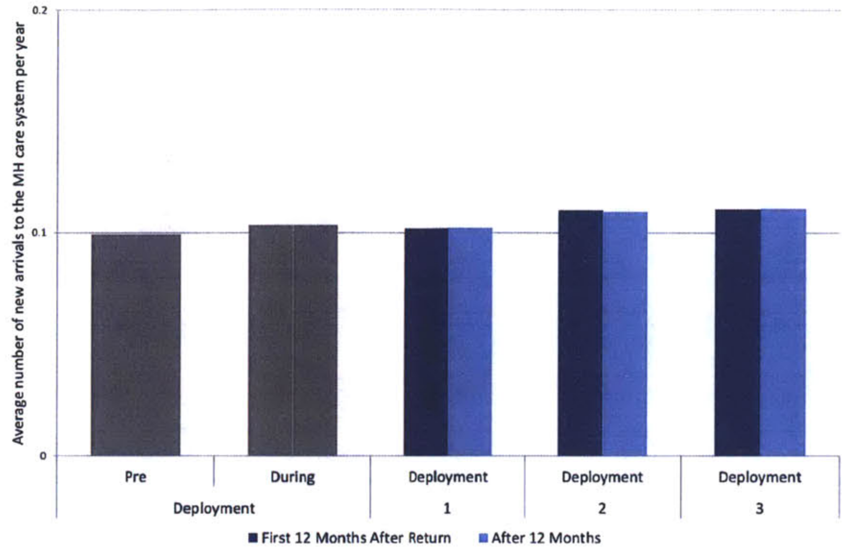
Compared to their predeployment care arrival rates, dependents of ADSMs entered care more often when their service member was deployed and after their service members return. This behavior was more pronounced in the Army, Air Force and Marine Corps. Dependents in these three services also began new episodes of care at higher rates following each subsequent deployment of their service member.

Figure 11: Arrivals to the System of Care by Dependents of Active Duty Soldiers



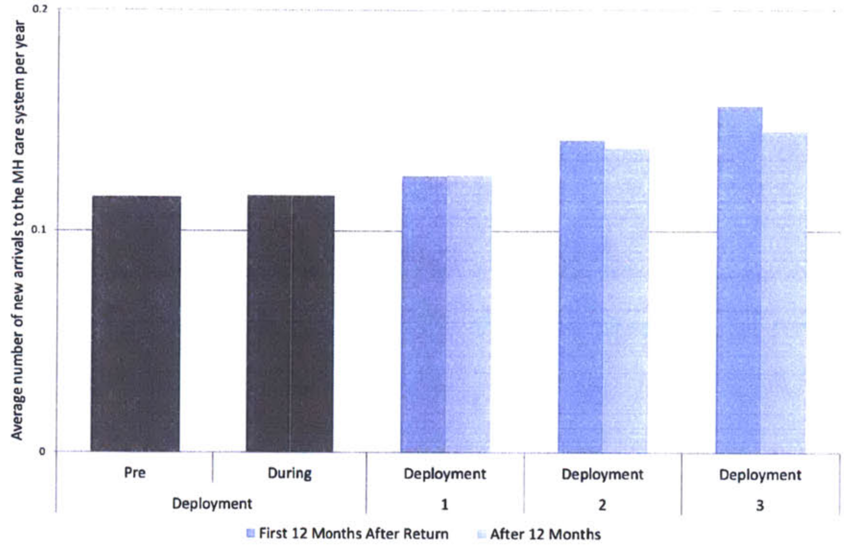
Army Families entered care at higher rates during their service member's deployments (+12%) than they did before their service member had ever deployed. After their SM's first deployment, they entered care at roughly the same rate. The year following subsequent deployments saw increasing numbers of new episodes (After First Deployment: +14%, Second Deployment: +23%, Third Deployment: +25%).

Figure 12: Arrivals to the System of Care by Dependents of Active Duty Sailors



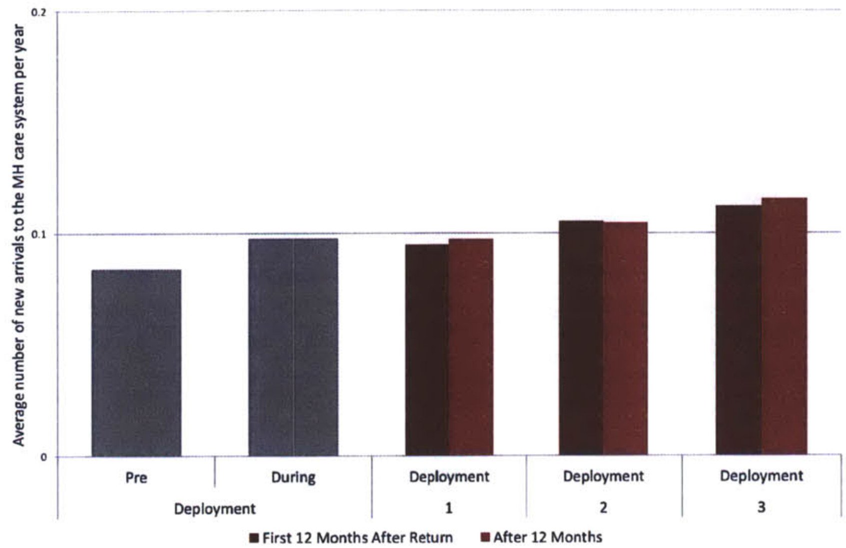
Navy Families entered care at roughly the same rate across the deployment cycle (During: +4%, First Deployment: +3%, Second Deployment:+ 11%, Third Deployment +11%).

Figure 13: Arrivals to the System of Care by Dependents of Active Duty Airmen



Air Force Families entered care at the same rate during their service member's deployments as they had before. After each return, from deployment, family members entered care at marginally higher rates (First Deployment: +8%, Second Deployment: +22%, Third Deployment: +36%).

Figure 14: Arrivals to the System of Care by Dependents of Active Duty Marines



Marine Corps Families entered care at a 16% higher rate during their service member's deployments than before. As with Army and Air Force families, arrivals to the system of care increased after each deployment (First Deployment: +13%, Second Deployment: +25%, Third Deployment: +34%).

4.3.2 Active Duty Service Members

The rest of this page intentionally left blank.

Army Figure 15a shows that soldiers entered care at drastically higher rates in the year following their return from theater than they did before ever deploying (1st Deployment: +104%, 2nd Deployment: +129%, 3rd Deployment: +123%). However, soldiers were comparatively less likely to enter care after this 12 month window had passed.

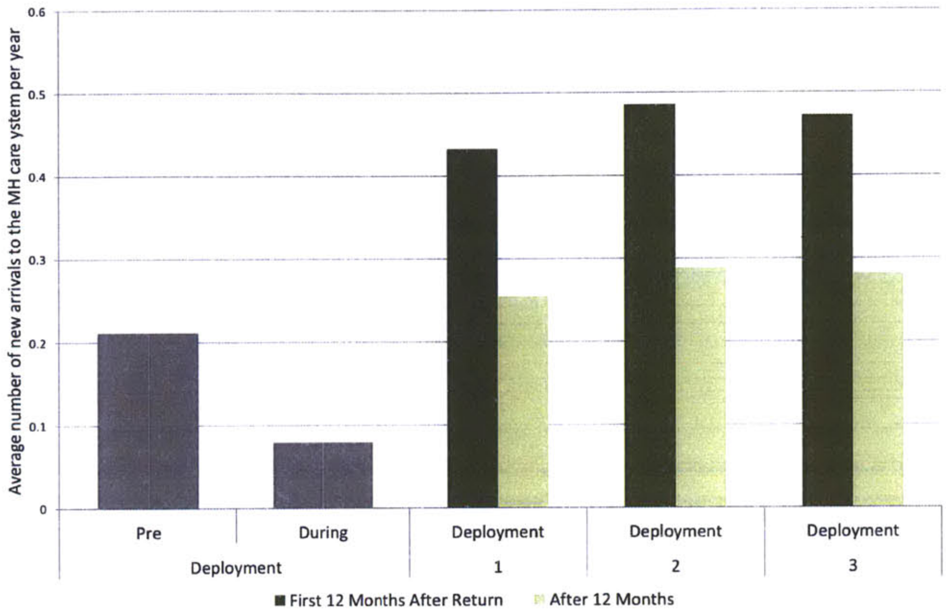
Soldiers often begin episodes immediately following their deployment, but this rate drops off sharply within the first 20 days . This spike may be due to routine post-deployment screenings (which were inconsistently implemented across installations), command directed mental health evaluations (CDMHEs) for service members whose behavior concerned their unit leaders, or an influx of patients who weren't able to get conventional care in theater. Likely, it is a mix of all these and more.

Rates rise between 60 and 120 days after return and gradually fall off to a steady state by about 240 days after return from deployment. This confirms anecdotal reports from several interviewees who reported that soldiers struggle with mental health issues about 90-100 days after deployment.

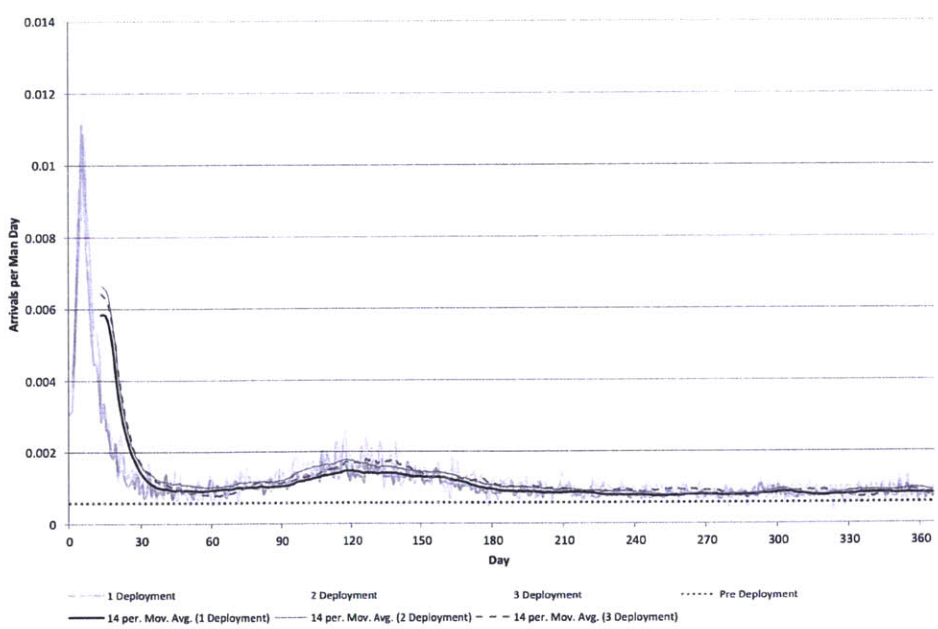
Arrival rates for service members were roughly the same regardless of the number of deployments the service member had been on.

Figure 15: Arrivals to the System of Care by Active Duty Soldiers

(a) Over multiple deployments



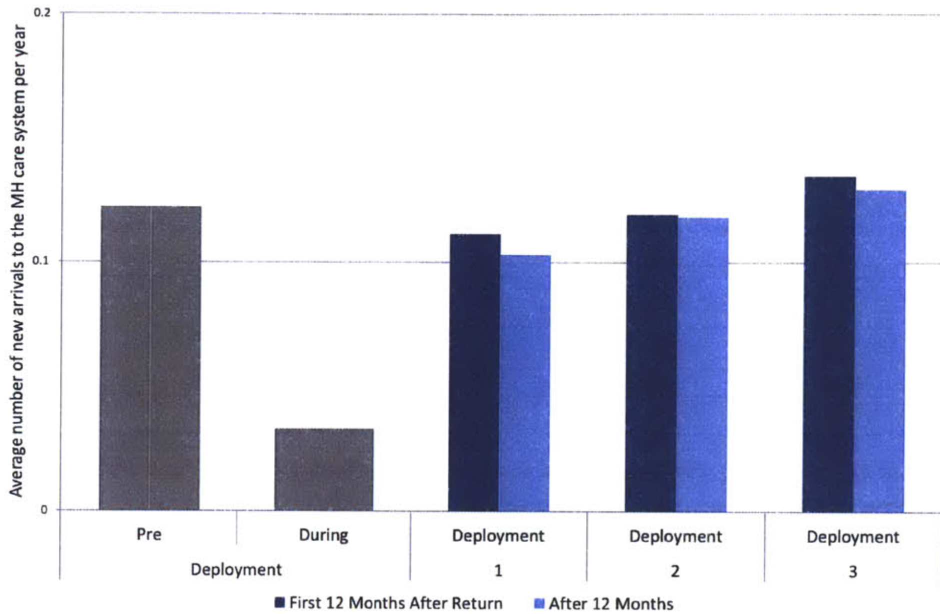
(b) In the 12 months after returning from a deployment

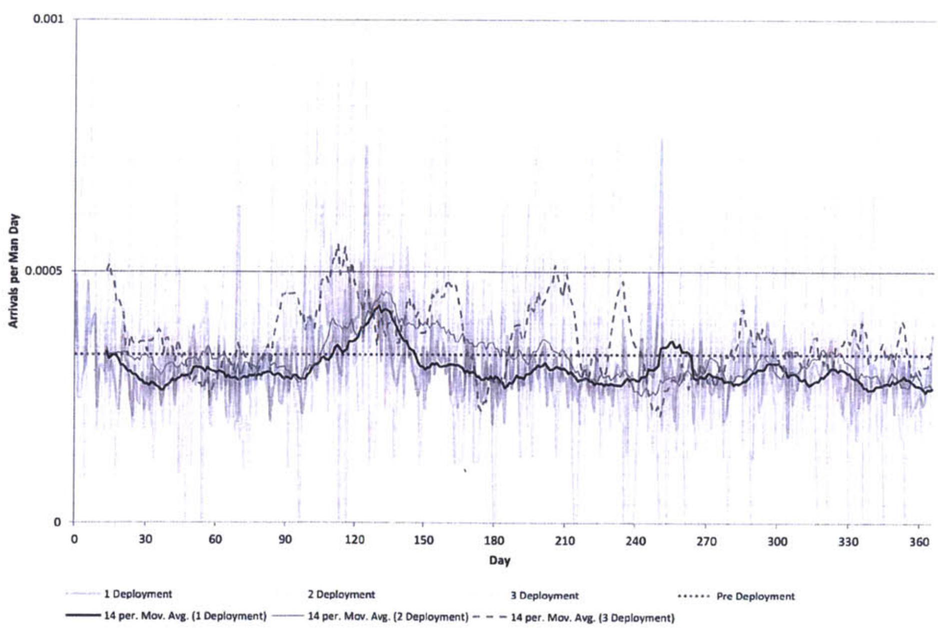


Navy Sailors entered care less frequently after their first and second deployments than they had before ever deploying (Figure 16a). After their third deployments, they saw only a small increase over the predeployment baseline rate (+10%). Looking at arrivals to the sytem of care over the first 12 months after return shows that arrival rates are fairly close to the predeployment baseline (Figure 17b). Any apparent trends after the third deployment should probably be discounted because of the small sample size.

Figure 16: Arrivals to the System of Care by Active Duty Sailors

(a) Over multiple deployments





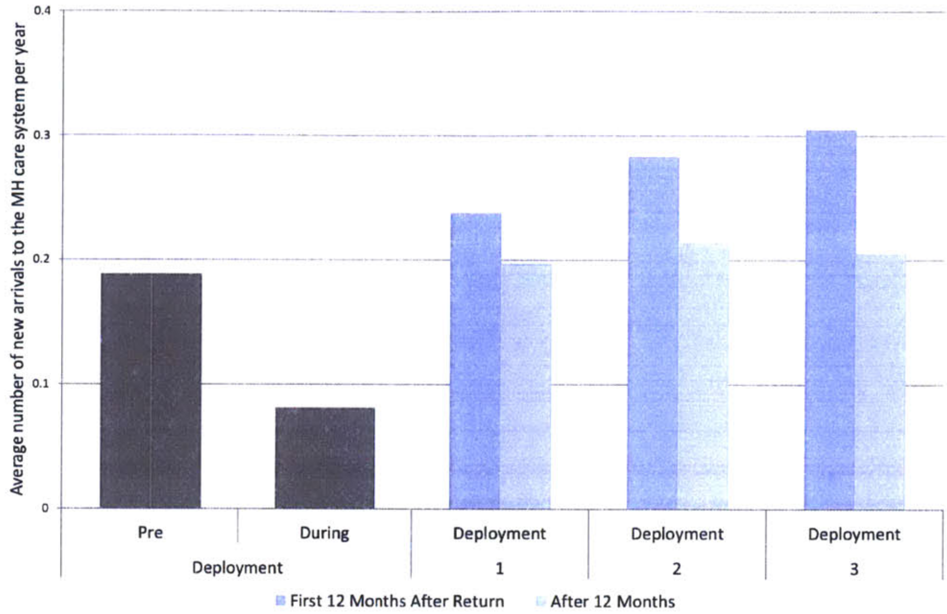
(b) In the 12 months after returning from a deployment

Air Force In the 12 months after deployments, airmen entered the system of care at rates much higher than before they deployed (Figure 17a). After their first return, they entered care at a rate 26% higher. Subsequent deployments were followed by arrival rates 50% higher and 62% higher than the predeployment baseline arrival rate. Like soldiers, they entered care at much higher rates in the 12 months after return than they did after that window.

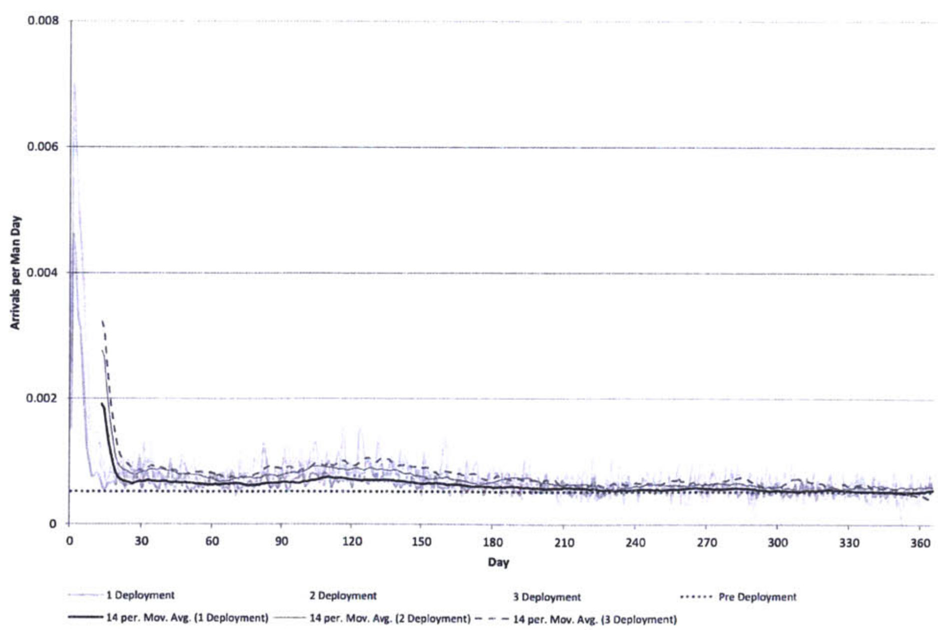
In the first 12 months, arrival follow a similar pattern to utilization (Figure 17b). There is however, one notable exception. Even though utilization (shown in 9b) drops below predeployment levels shortly after return, arrival rates to the system do not.

Figure 17: Arrivals to the System of Care by Active Duty Airmen

(a) Over multiple deployments



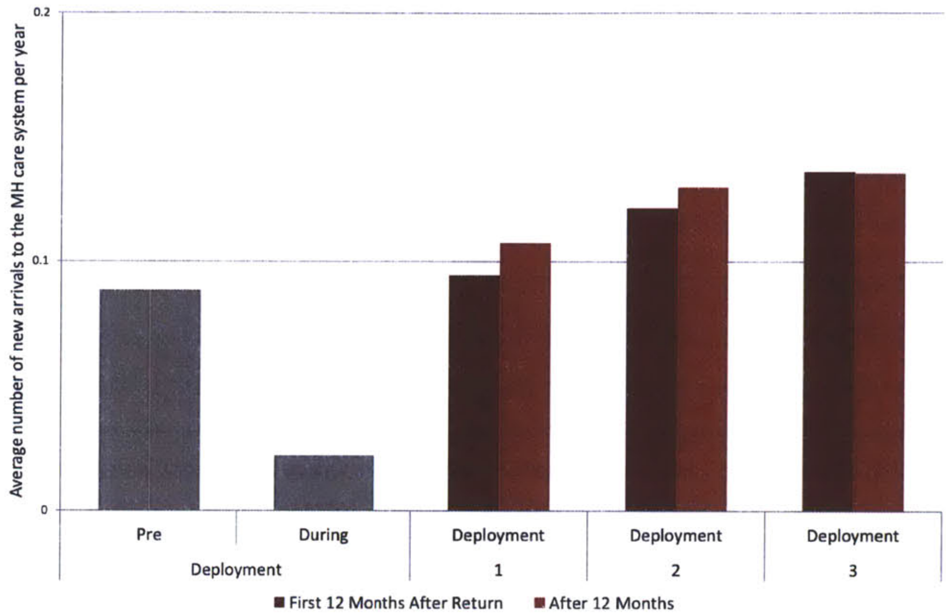
(b) In the 12 months after returning from a deployment

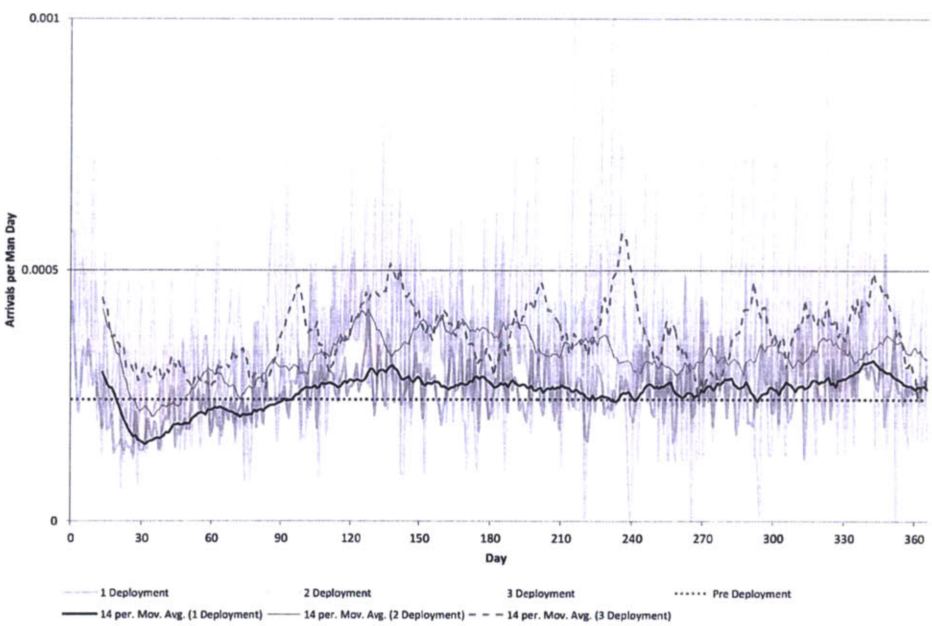


Marine Corps Figure 18a shows that each deployment was followed by a higher rate of care arrivals than before deploying or after preceeding deployments (First Deployment: +7%, Second Deployment: +37%, Third Deployment:+ 54%). Figure 19b shows a small spike right away, followed by a dip in arrivals, followed by relatively consistent arrivals over the duration of the year. The arrival rates for returning Marines were more constant than their utilization patterns.

Figure 18: Arrivals to the System of Care by Active Duty Marines

(a) Over multiple deployments





(b) In the 12 months after returning from a deployment

4.3.3 Utilization Lags Arrivals

As expected, utilization lags arrivals. The slow growth in utilization over the first 12 months post-deployment for the Navy and Marine Corps population are the result of relatively constant arrival processes. Patients who arrive generally recieve a few encounters over the course of a few weeks, so it takes several weeks to build a steady state sized case load. Figures 19 through 22 show these lagging trends for each service.

In the Army (Figure 19), we can see that care utilization is the lagging effect of service members entering care and visiting their provider several times over a period of weeks. Although new encounters reach a constant rate by day 210, care consumption does not plateau for approximately another 60 days. New encounters in the Navy begin at nearly the same rate as before deploying for the duration of the first year after return (Figure 20). Again, we see that this pattern causes a lagging result in total care recieved. Despite regular arrivals to the system, the caseload of Active Duty service members doesn't reach steady state until about four months after return. During the first few months after Air Force returns, arrival rates show a slow decrease and utilization increases. Like Sailors, Marines see a relatively constant arrival rate, which drives the gradual increase and subsequent plateau of their utilization rate (Figure 22).

Figure 19: Army Utilization Lags Arrivals

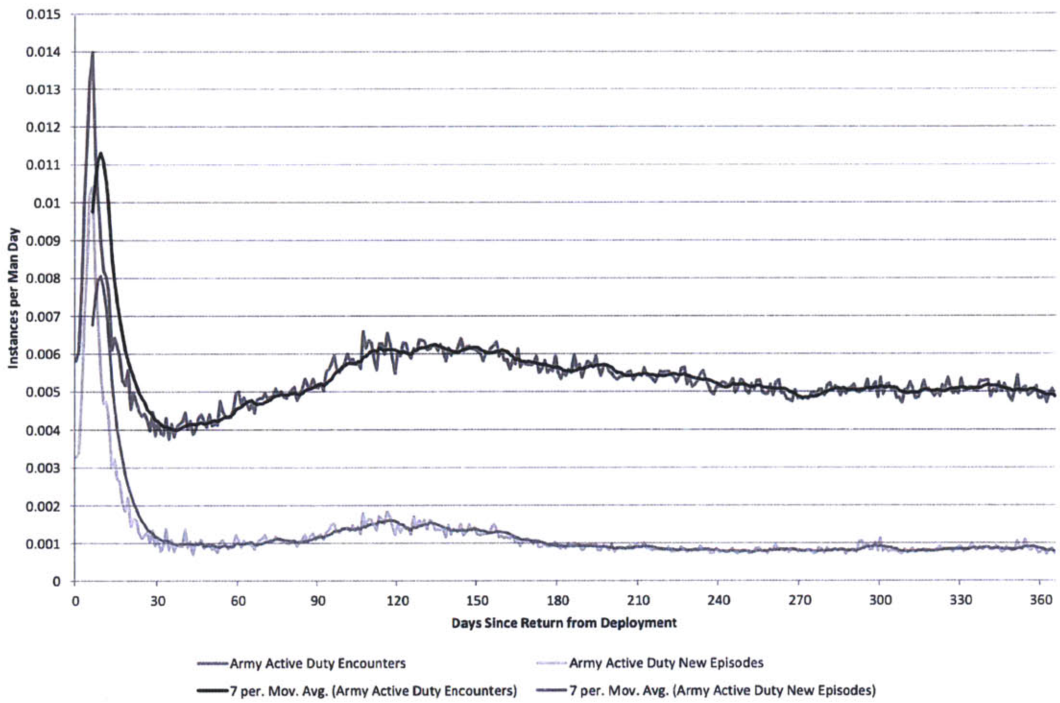


Figure 20: Navy Utilization Lags Arrivals

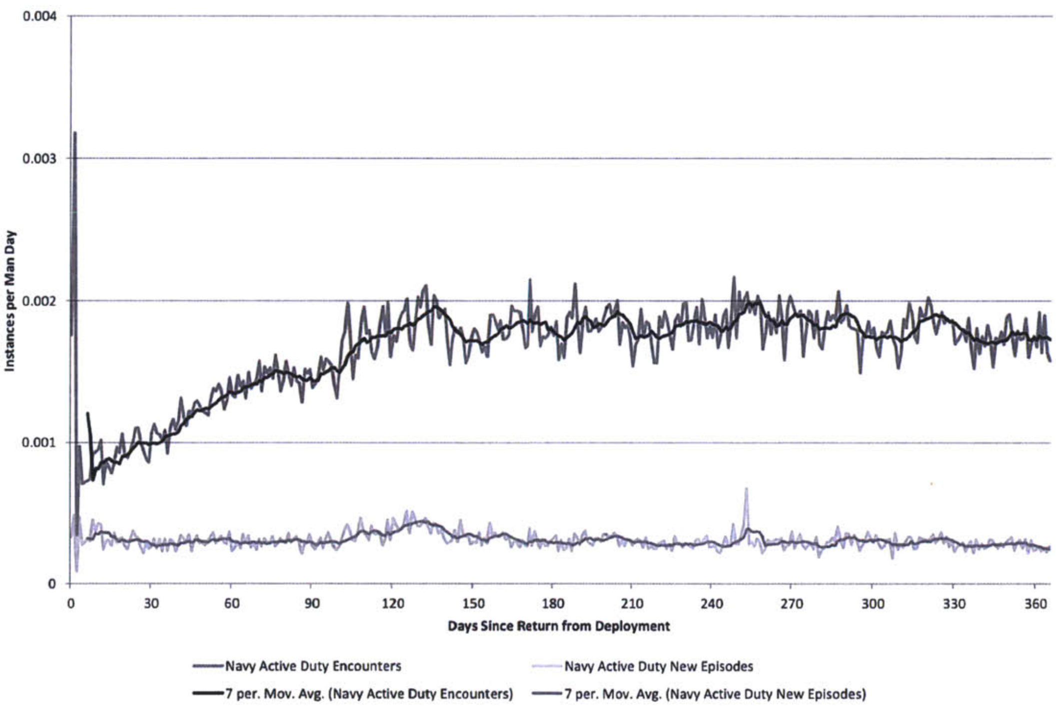


Figure 21: Air Force Utilization Lags Arrivals

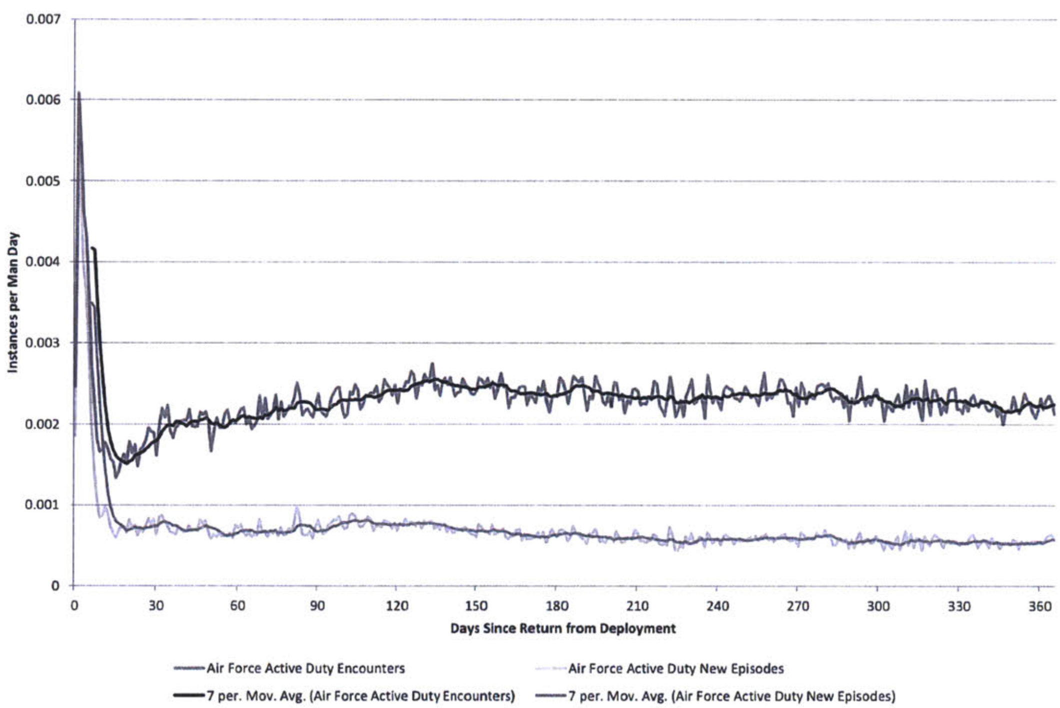
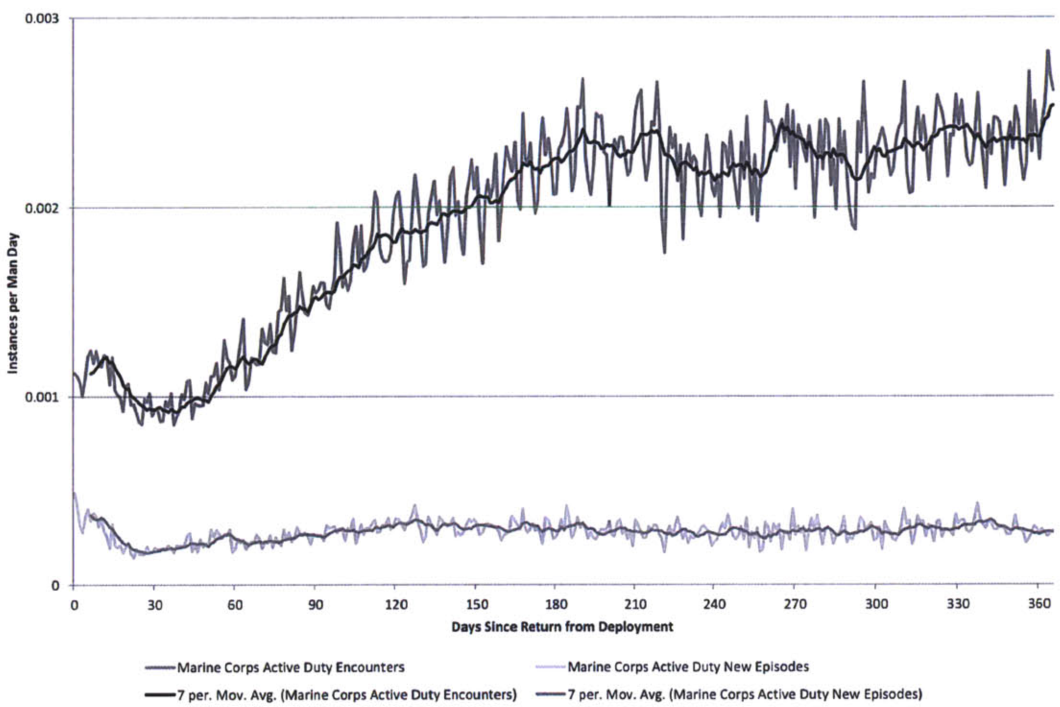


Figure 22: Marine Corps Utilization Lags Arrivals

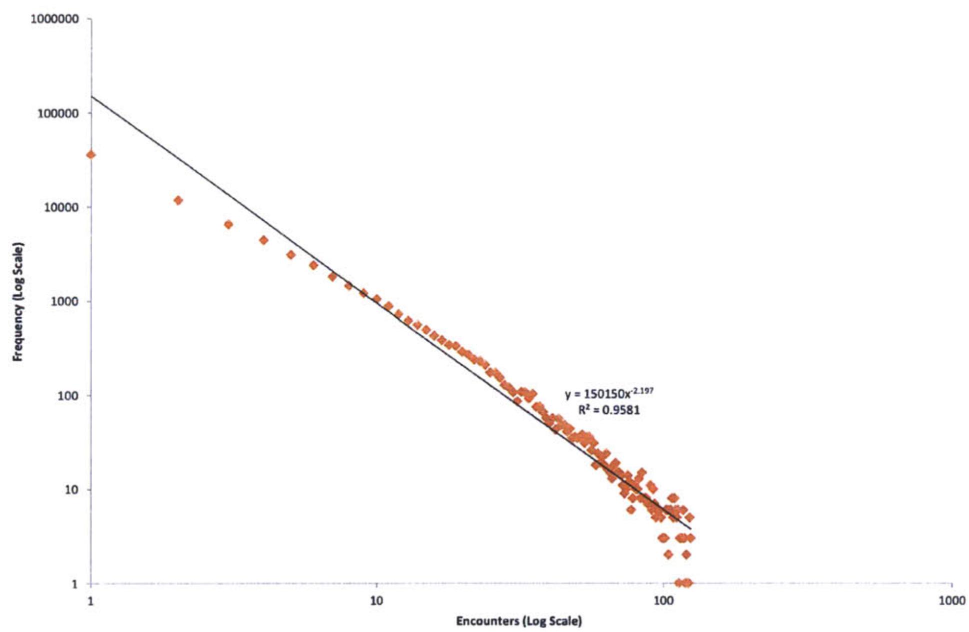


4.4 Characteristics of an Episode of Care

Care utilization per patient is sometimes modeled as a constant value for a given diagnosis [Harris et al., 2010]. When computing aggregate demand for many beneficiaries over a long period of time, this approach is suitable. When simulating the system at a daily granularity, it is useful to understand the probabilistic length of an encounter, since constant length and variable length service times cause systems to behave in substantially different ways.

Beneficiaries who see a mental health provider are most likely to recieve just a single encounter. Overall, the distribution of encounters per episode follows a power law distribution relatively neatly (Figure 23). Clearly, there is a wide variation in utilization per episode. Even though only about 5% of patients have epidodes lasting more than 20 encounters, they utilize 40% of care. Conversely, 45% of arrivals recieve just one encounter but they consume only 9% of encounters[13].

Figure 23: Log-Log Plot of # of Encounters/Episode by Frequency



4.5 Population and Demand Projections

A similar algorithm to the one above was used to calculate the actual daily population of each of six Army installations and two Marine Corps installations. These daily populations were broken down

[13]The analysis presented here does not cover the complete data set. It covers approximately 78,000 episodes of care which account for some 398,000 encounters.

into unique values of B, P, and S. Using the daily population, we can determine the population of a site on any given day. Using these very specific population counts and the arrival rates calculated in Section 4.3, it is possible to project daily demand at each installation according to the equation:

$DailyDemand = \sum_{B,P,S}(DiagnosisRate_{B,P,S} * DailyPopulation_{B,P,S})$

A full account of this methodology and the source code for determining the rates is presented in B. The projections below cover the same time window used in the simulations in Sections 6 and 7. Populations and demands were also calculated for six months preceeding this window. Those rates were used to "warm up" the simulation (not shown).

Figures 24 through 29 show the populations at six Army installations. Installation Alpha through Echo each see at least one major "surge" in demand following the return of a large combat unit.

For example, at site Alpha (Figure 24), about 12,000 service members return from deployment starting at about day 240. This causes the rate of arrivals to go from about fifty per day to near 85 at the peak. Arrival rates stay elevated (about 20% higher than before) for several months until another large group of service members deploys, lowing the population back to about 95,000 and bringing demand back down.

Site Foxtrot (Figure 29), however, does not see substantial surges in demand. Unlike the others, site Foxtrot is not a power projection platform, so large groups of soldiers stationed there do not deploy.

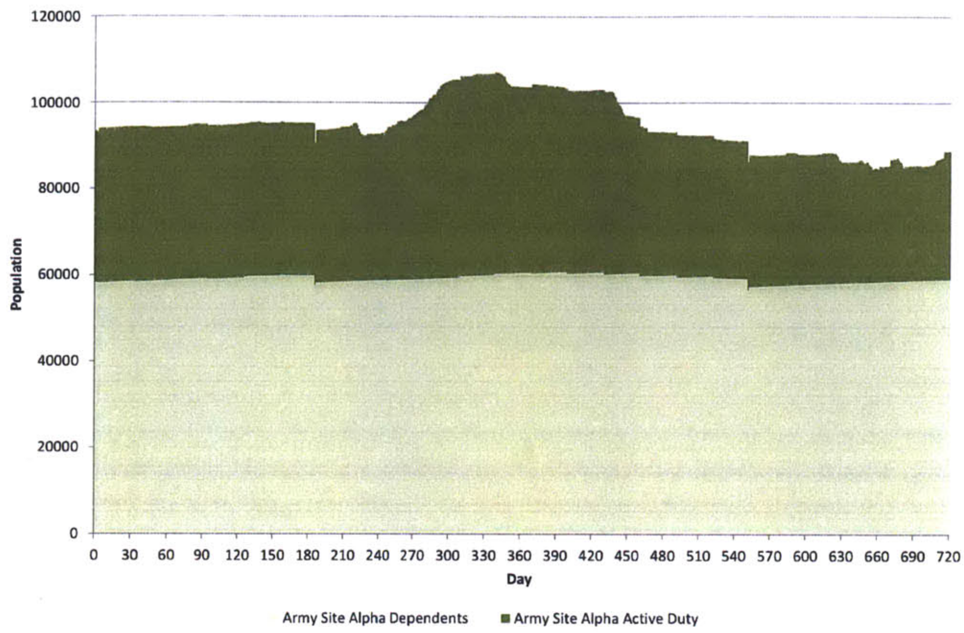
Figure 30 shows the combined populations and arrivals of all six sites. Overall, both the population and the arrival rates are more steady than at any of the power projection platforms (Sites Alpha, Bravo, Charlie, Delta and Echo).

Marine installations Kilo and Lima (shown in Figures 31 and 32) show a very different pattern. Each is home to a Marine Expeditionary Force and sends service members on frequent deployments. But, because Marines (according to historical data) did not enter care at particular intervals, they generated much more consistent demand patterns.

Because Army installations see what appears to be a more challenging demand patterns, and because the qualitative work informing this demand generation methodology relied on interviews with Army stakeholders (see Section 8), only the Army sites shown here are simulated in the following sections.

Figure 24: Populations and Projected Arrivals: Army Site Alpha

(a) Population



(b) Arrivals to System of Care

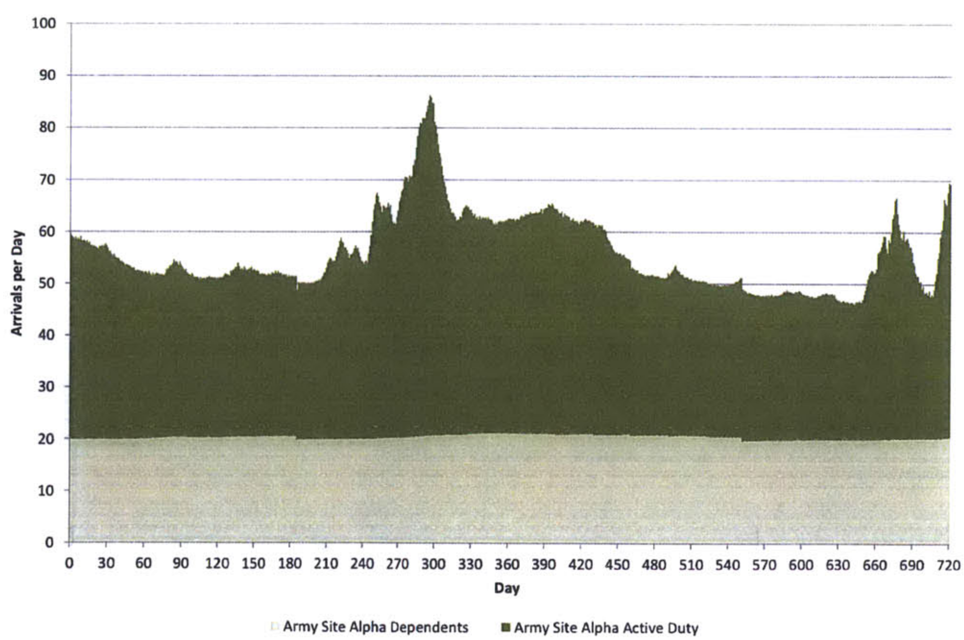
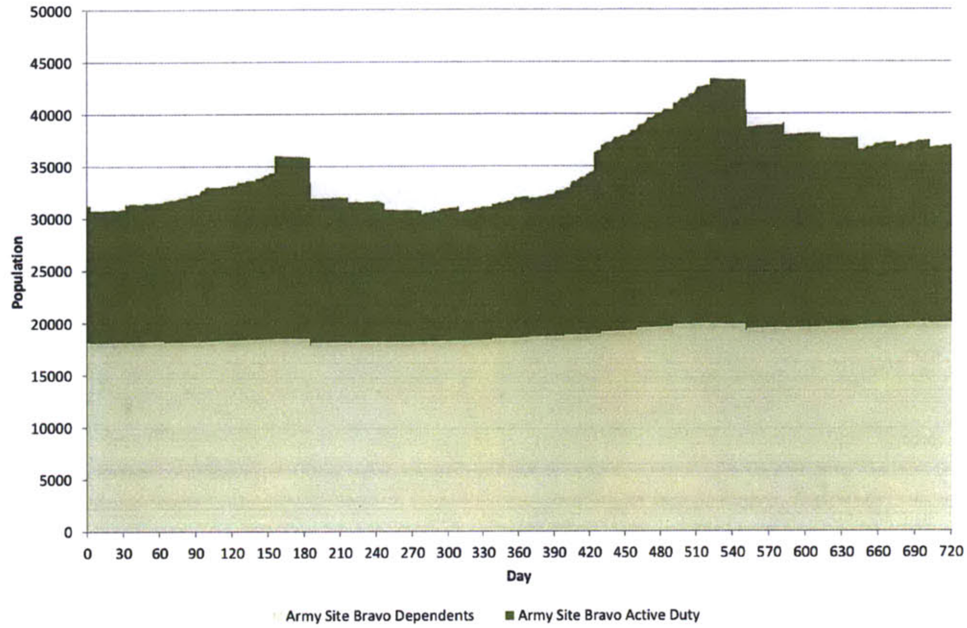
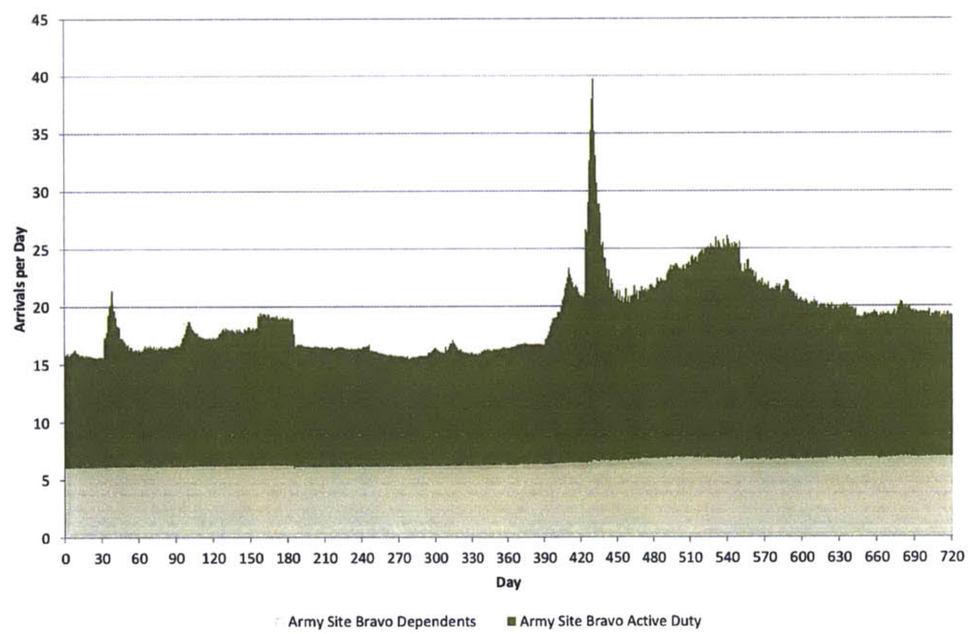


Figure 25: Populations and Projected Arrivals: Army Site Bravo

(a) Population

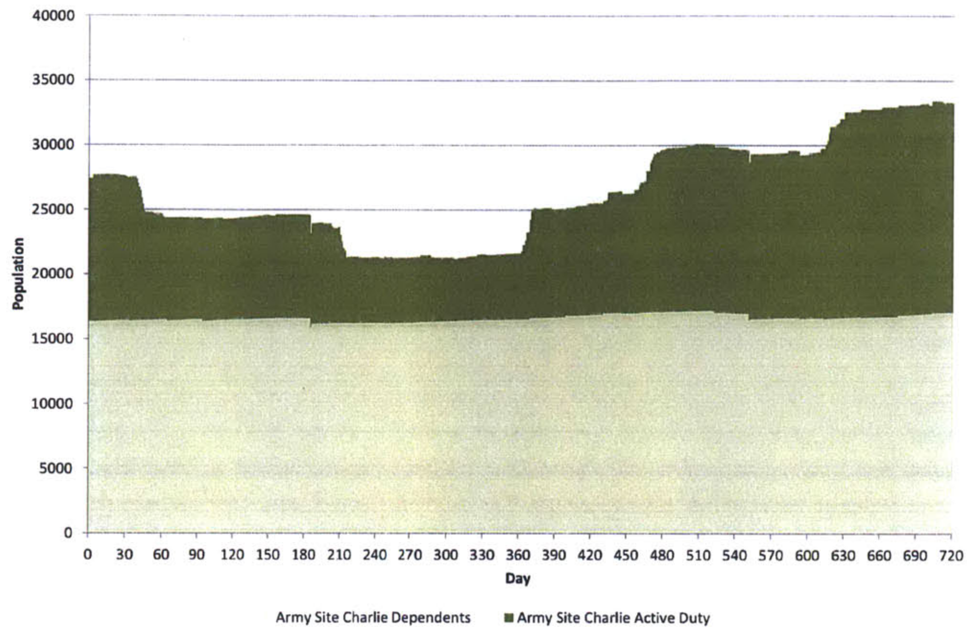


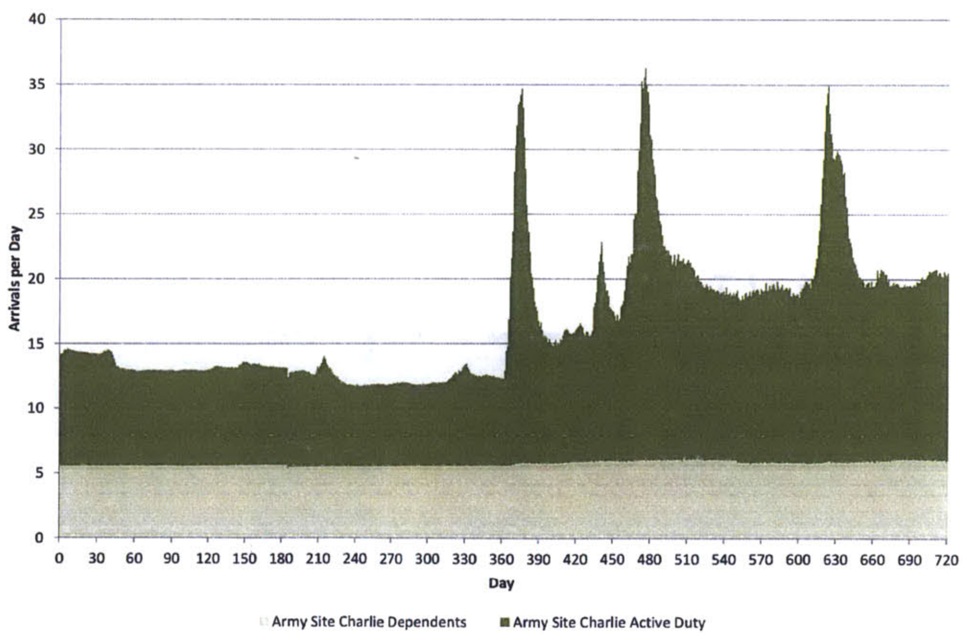


(b) Arrivals to System of Care

Figure 26: Populations and Projected Arrivals: Army Site Charlie

(a) Population

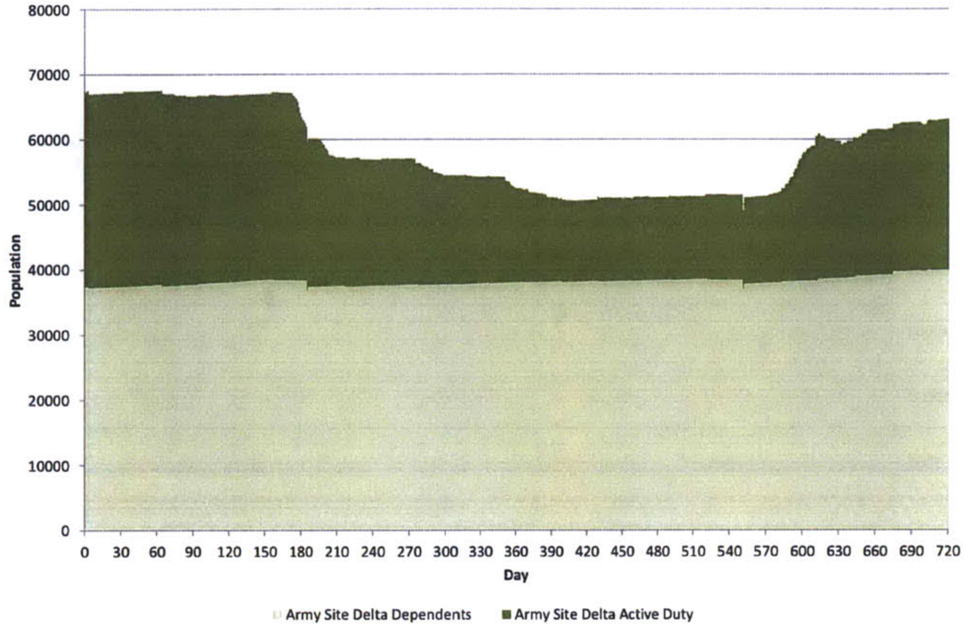


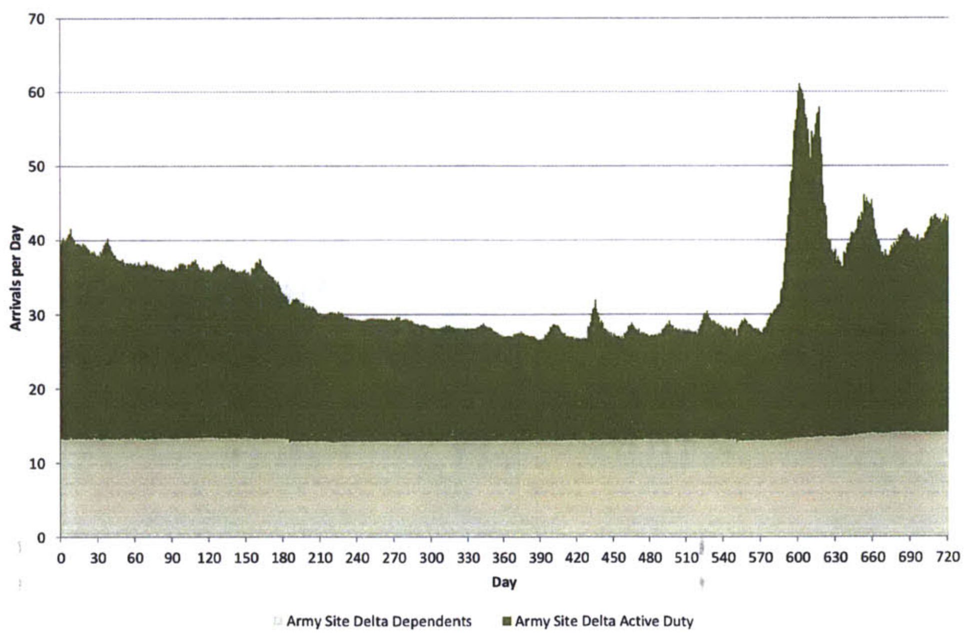


(b) Arrivals to System of Care

Figure 27: Populations and Projected Arrivals: Army Site Delta

(a) Population

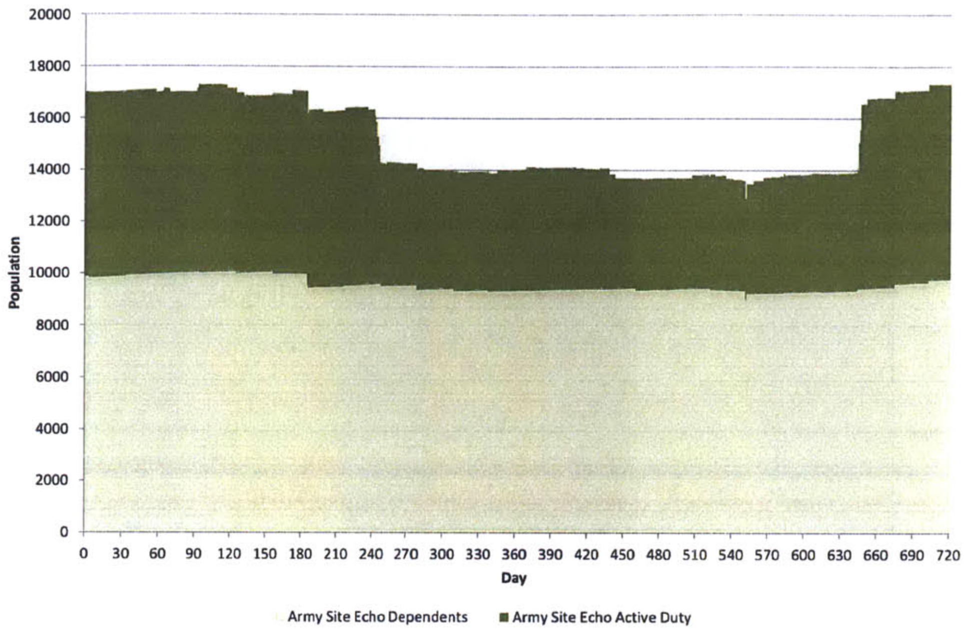


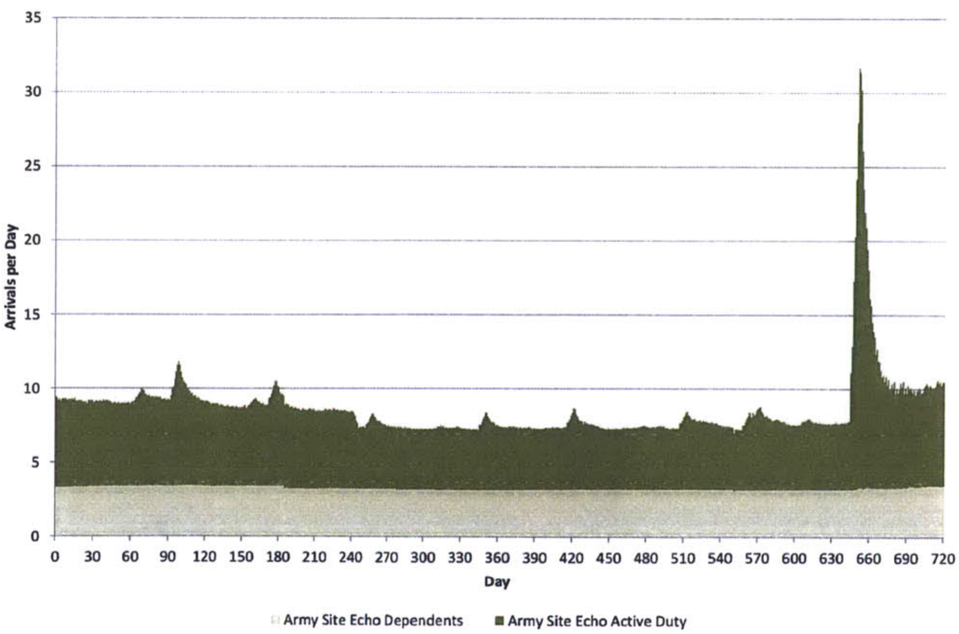


(b) Arrivals to System of Care

Figure 28: Populations and Projected Arrivals: Army Site Echo

(a) Population

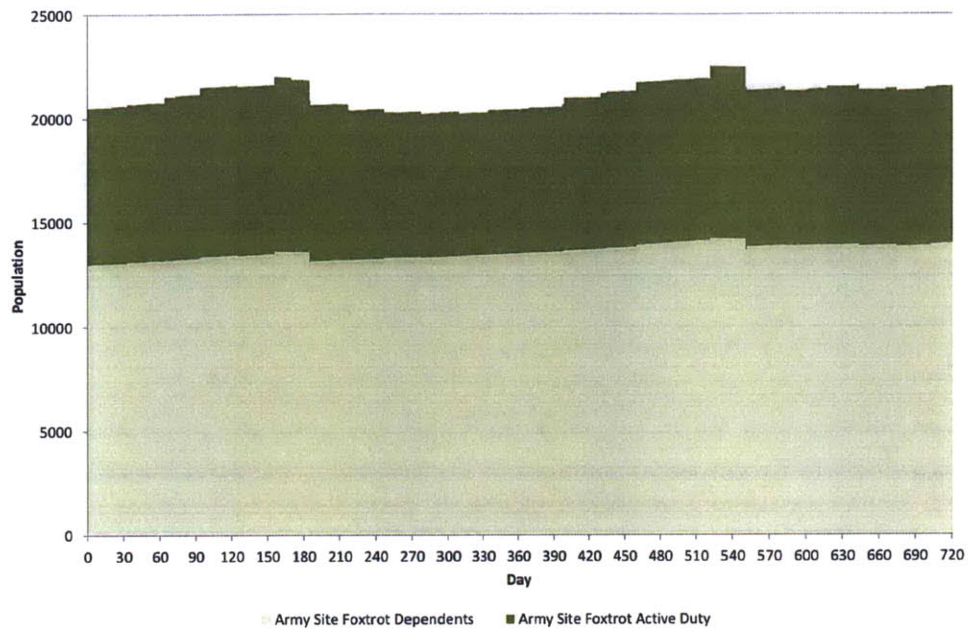


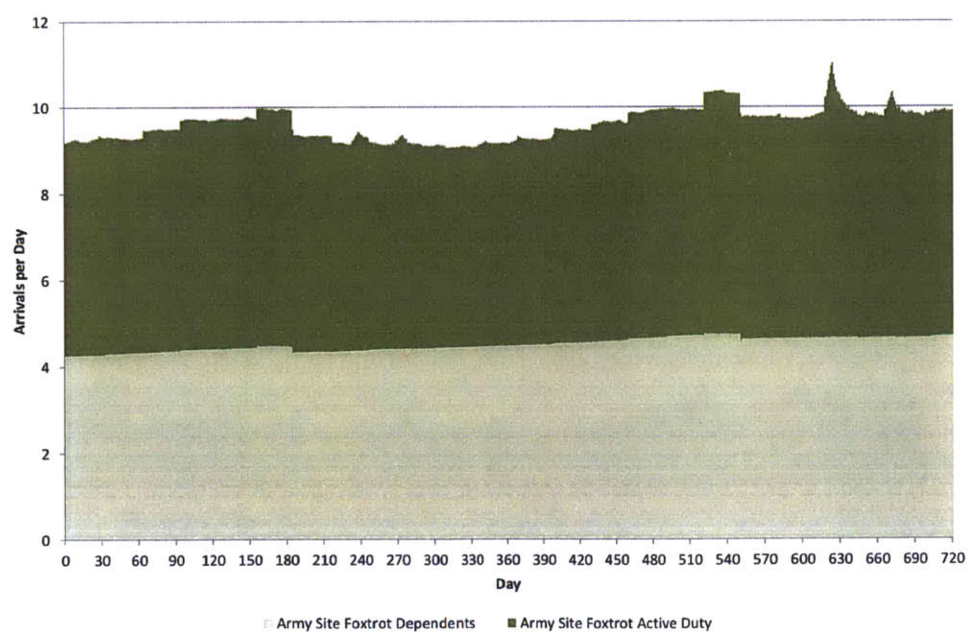


(b) Arrivals to System of Care

Figure 29: Populations and Projected Arrivals: Army Site Foxtrot

(a) Population

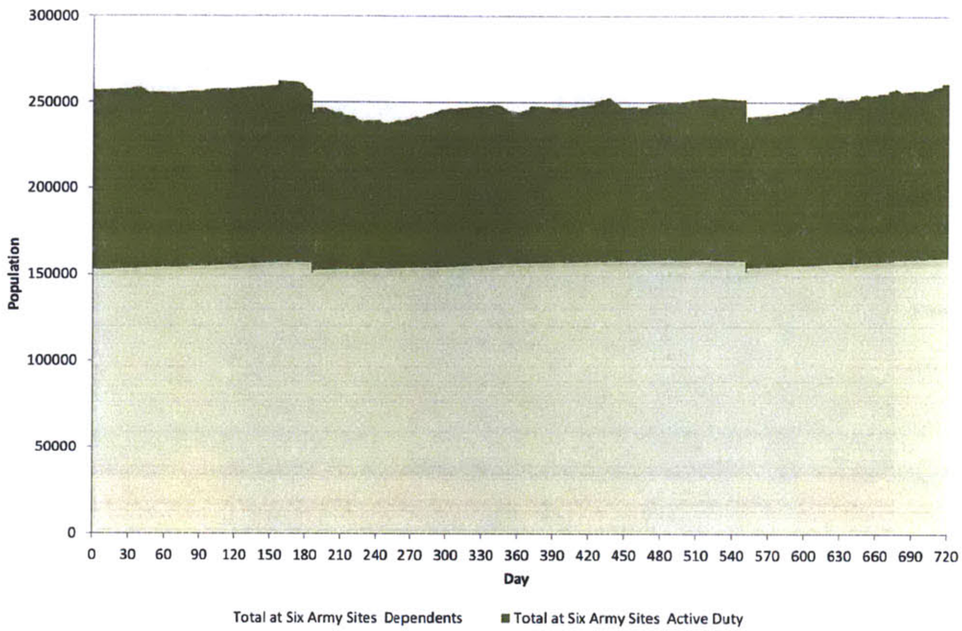




(b) Arrivals to System of Care

Figure 30: Populations and Projected Arrivals: Six Army Installations Combined

(a) Population



(b) Arrivals to System of Care

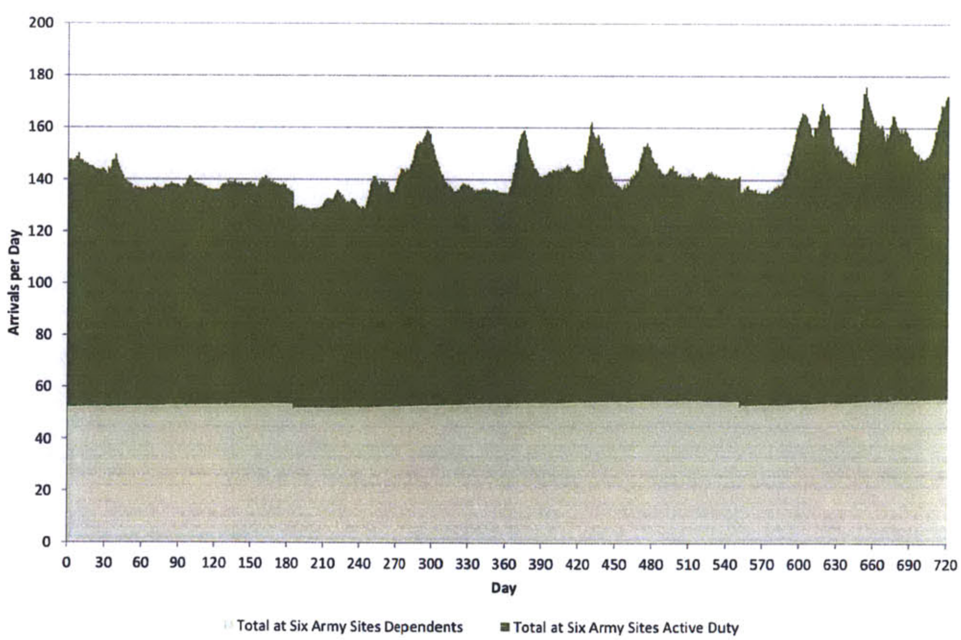
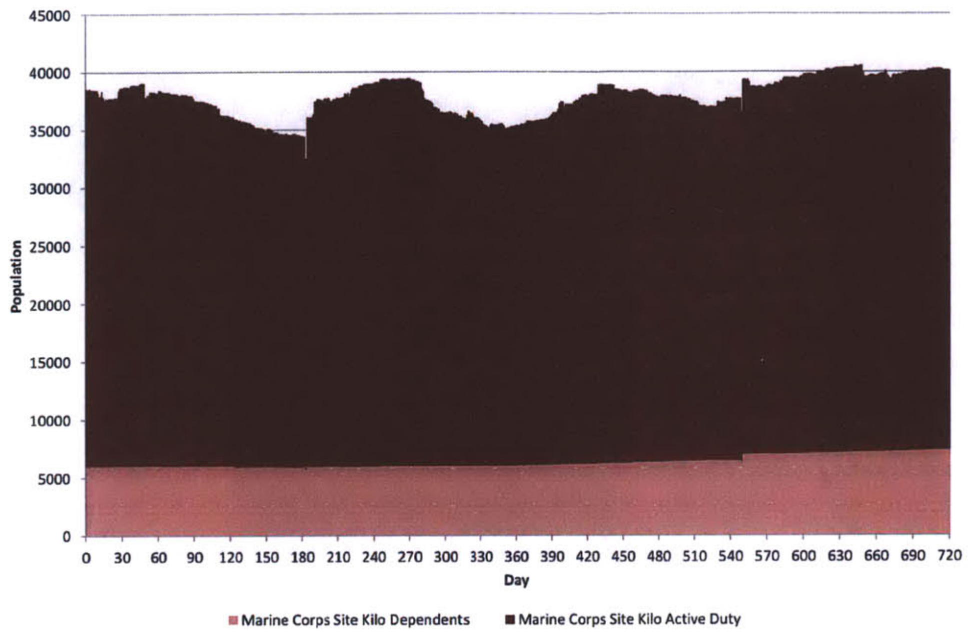


Figure 31: Populations and Projected Arrivals: Marine Corps Site Kilo

(a) Population



(b) Arrivals to System of Care

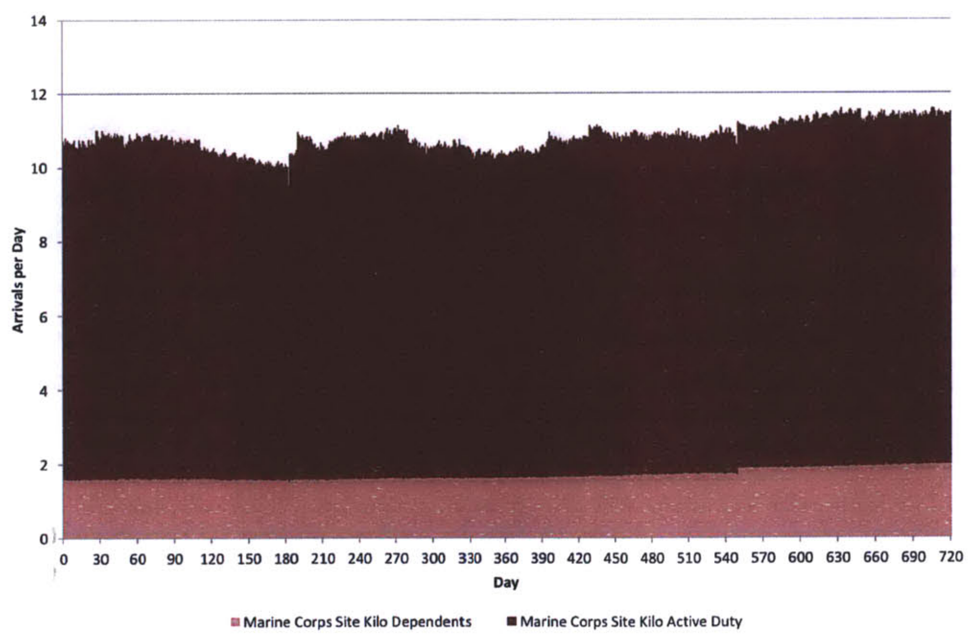
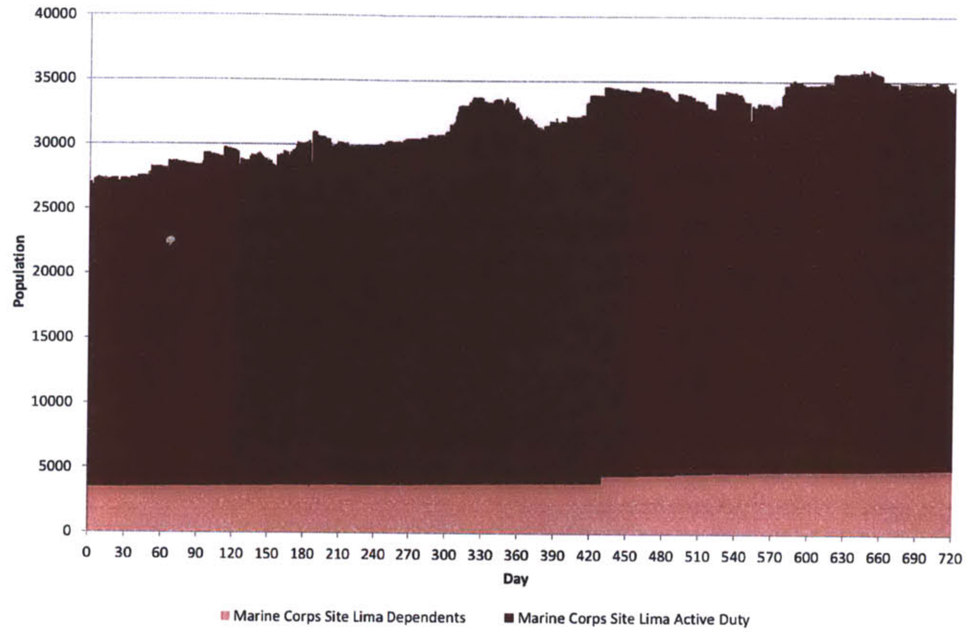
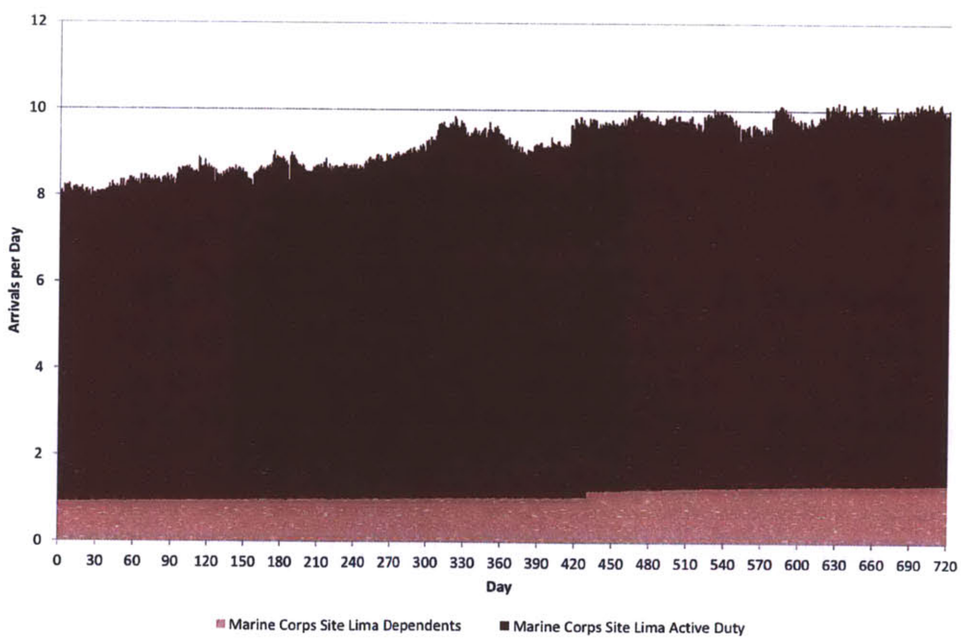


Figure 32: Populations and Projected Arrivals: Marine Corps Site Lima

(a) Population



(b) Arrivals to System of Care



5 Simulating the System of Care

5.1 Methodology

5.1.1 Why Simulation?

Simulation affords us the ability to examine potential changes to the military's system of care that are difficult to analyze using other methods. The lag-times associated with training, deployment, exposure to traumatic events, affliction with PTSD, treatment and eventual health outcomes limit the utility of real experiments. Furthermore, there is considerable variation between subpopulations in the military and creating convenient control groups is difficult. Even if these mechanical issues could be overcome, experimenting on the system of mental health care is fraught with concerns for patient well being.

Retrospective data analysis have generated valuable insight into the epidemiology of Post Traumatic Stress Disorder and other conditions within the military, but have only limited capability to evaluate the effects of changes to the system of care.

5.1.2 Why Discrete Event Simulation?

DES is a natural methodology for understanding complex outpatient care centers [Berg and Denton, 2012] like those operated by the military. A Discrete Event Simulation was chosen over simpler methods because it can evaluate an arbitrary pattern of demand and create compelling visualizations which are useful for communicating with policy makers and other stakeholders [Utley and Worthington, 2012].

Ordinarily, standard queueing models are useful for evaluating the effective throughput of a service system and expected waiting times. The solutions to such formulations are often closed form, computationally simple. Variation from expected results can be robustly defined. Unfortunately, such an approach often relies on a simply defined demand function (e.g. poisson/exponential or constant rate). The analysis in Section 4 shows that demand in this system varies substantially based on systemic behaviors and can't be reasonably approximated by a simple probabilistic arrival patterns.

5.2 Simulation Design

Using the demand projections from Section 4.5, the simulation generates patient arrivals. Patients are assigned a set of attributes including beneficiary type and need for care. Patients try to seek care from their home station's MTF and may be accepted there, referred for care in the direct care system via telehealth or referred out to the home station's purchased care system. Once they enter a system of care, patients recieve encounters until their need has been met. Each simulated day, the simulation records key metrics which are later used to compare system performance under each scenario.

5.2.1 Simulation Inputs

The simulation accepts three main inputs: A file which describes the arrival rates of new patients, two values describing the topology of the TH network, and the maximum case load size of each MTF.

Arrival Pattern File The input file contains patient arrival rates[14]. For each day, this file contains a list of patient arrivals. For each patient, the file specifies the patients' home station and his or her beneficiary type: either Active Duty or Dependent. Upon arrival to the simulation, patients are assigned a need for care, expressed in the number of encounters they need before they leave the system. This number is randomly selected from the distribution presented in Section 4.4[15]. All simulated patients arrive at their home MTF and are sorted into one of three care systems (see the flowchart in Figure 33).

MTF Care Capacities Each MTF has a constant capacity for care delivery. For all scenarios in this thesis, this is set at 50% of the average demand placed on an MTF over the duration of the simulation.

TH Network A binary input "TH On?", helps control patient flow. When TH is turned off, patients will automatically be rejected from any TH system.

Each simulation run also requires a caseload size for the TH Cell. If "TH On?" is false, this is 0. Otherwise, the user specifies the size of the TH Cell. If the user sets it to zero (meaning there is no telehealth cell), patients can only receive TH from other installations which have spare capacity. If it is set to a non zero value, patients will queue for the TH provider cell and fill the cell's appointment book before trying to seek care from another installation.

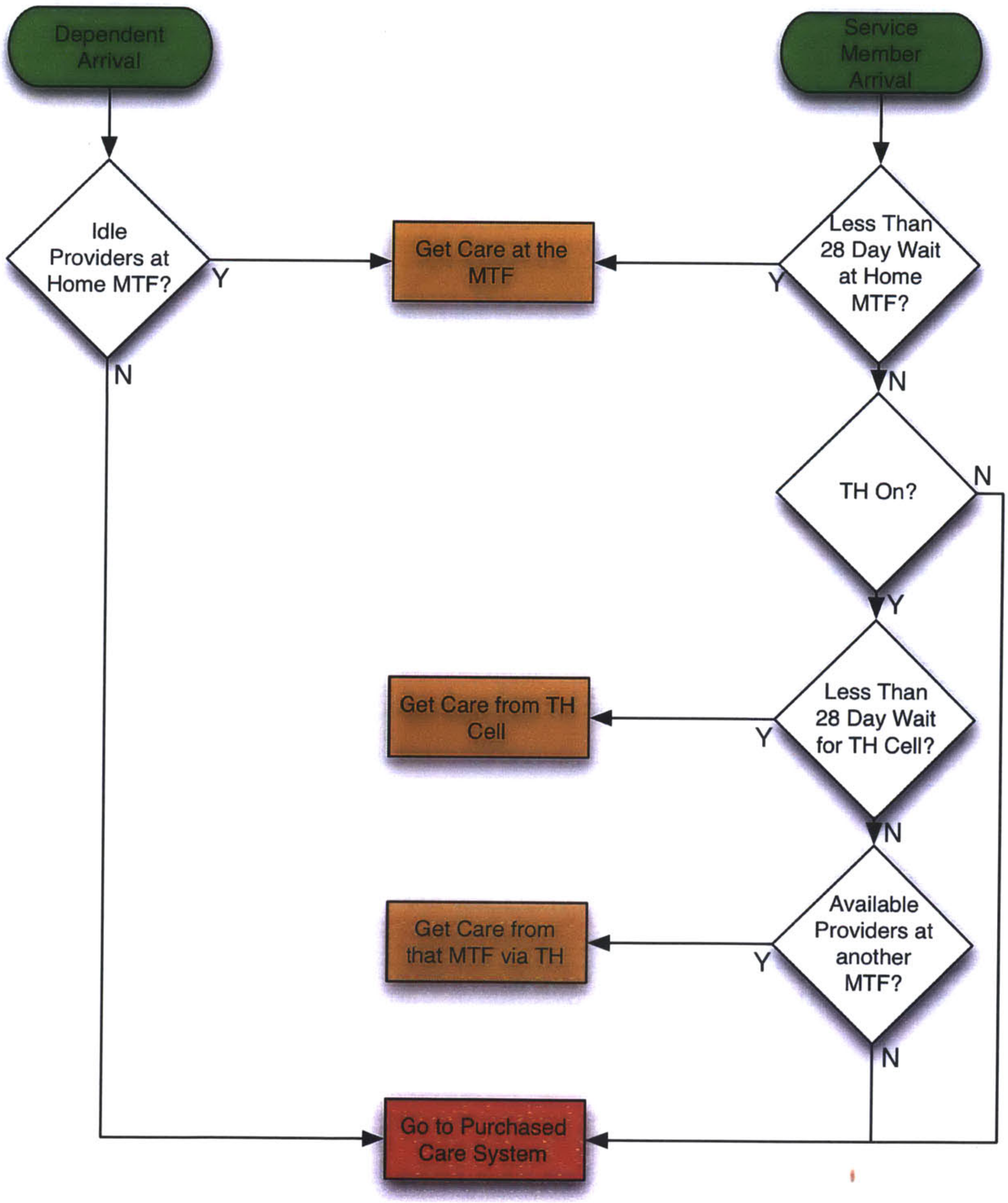
5.2.2 Sorting Patients to Different Systems of Care

Federal regulations specify that ADSMs take priority over dependents. This simulation guides patient flow accordingly.

[14] Arrivals in this file reflect a poisson arrival process for each 24 hour period with an arrival rate equal to the expected arrival rate for that day. As a result, there will be some variation in arrivals from day to day even when the expected number of arrivals is the same, but the overall pattern of arrival trends persists.

[15] Number of encounters is selected from the empirical data, not from the power law curve fit. This makes the methodology robust to any systemic variations, such as the prevalence of screenings or the 8 encounters family members are automatically authorized, which might cause a deviation from a "clean" distribution.

Figure 33: Sorting New Arrivals into Care Systems



Dependents If the patient is a dependent, and their home MTF has idle capacity, then they are served immediately (idle providers imply that there is no line). Otherwise, dependents are immediately sent to the purchased care system.

ADSMs If the patient is an ADSM, they will either enter direct care at their MTF immediately (if there are idle providers) or they will enter the MTF's queue if the wait for the next appointment is less than 28 days. If the MTF can't accept the service member in the next 28 days, they next try to get a TH appointment.

If there are providers sitting idle in a TH cell or if the wait to be seen by the TH cell is less than 28 days, they enter the TH cell's system of care. If they can't get in to the TH cell or make an appointment, they see if there are any installations with idle capacity or installations which are serving family members[16]. If there is an installation that can accept the service member immediately, they get care from that installation's providers via TH. Service members will not wait in line at distant installations since we assume an installation with a backlog will not accept increases to its patient load.

If the service member can not get direct care at their own MTF or via telehealth, they are sent to the purchased care network.

5.2.3 Care provision process

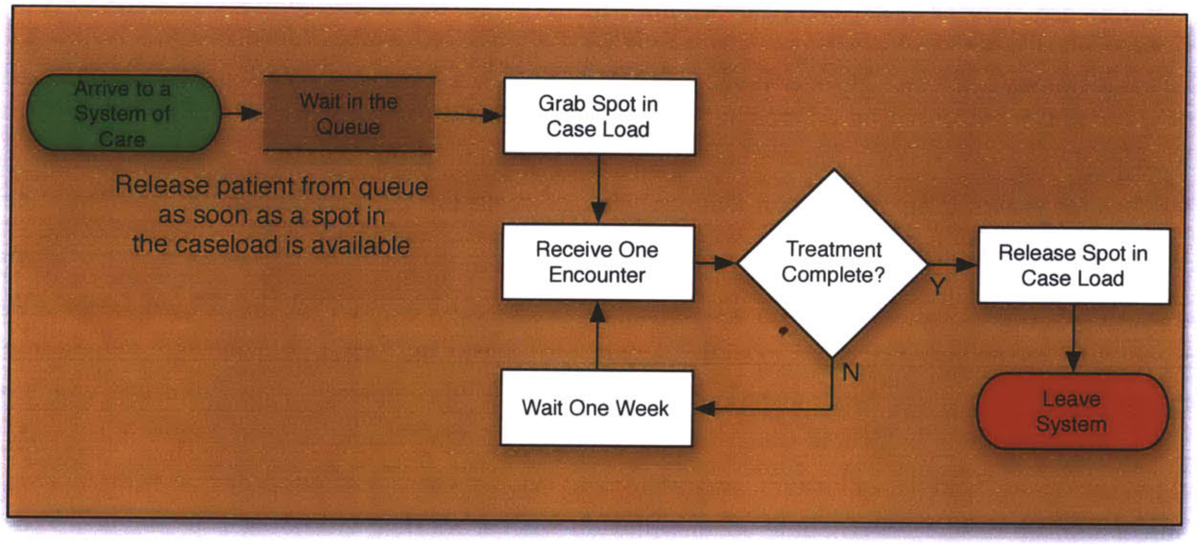
Care provision is fundamentally a two step process:

1. *Receiving the first encounter:* Patients can wait up to 28 days before beginning care. MHS sets different access to care standards for the very first encounter (7 days). In some cases, this means that patients get an immediate assessment, then wait up to 28 days for follow-on care [Scott and Srinivasan, 2012]. Because of this, the simulation assumes that the effective backlog a site can keep is 28 days, not 7.
   In reality, patients enter the system through any of the set of gate-keepers described on page 17. Patients may receive immediate care if they present in the Emergency Room or walk-in to a clinic, or they may make an appointment for a future date. Typically, all non emergency encounters will consist of an assessment and the assessor will assign the patient to a provider commensurate with their need (complex diagnoses requiring medication warrant a psychiatrist, for example).

2. *Receiving follow-on encounters:* Patients will begin follow on treatment and make follow on appointments according to the provider's availability. Clinical Practice Guidelines usually prescribe one encounter per week for the duration of care. In reality, this frequency varies

[16]The simulation reflects the regulations set forth in 32 CFR 199.17 and assumes it will continue to apply to telehealth. Therefore, any ADSM takes priority over local dependents in the simulation..

Figure 34: Care Provision Process in Direct Care Systems



based on severity and provider caseload. In this simulation, patients receive exactly one encounter per week.

Direct Care Systems All direct care systems (MTFs and TH cells) behave similarly. A representative flowchart of care within a system is presented in Figure on the current page. Patients wait in a line which represents the system's appointment book. Each system has a finite case load. If a spot is available (meaning that a provider is idle), the patient enters care. If a spot is not available, patients wait in the first come, first serve queue until other patients finish receiving care. Once patients enter, they receive care at a rate of one encounter per week[17]. Each day, the simulation checks to see if they are done with their treatment. If so, they leave the system of care, making room for another patient from the queue.

Each day, the simulation counts the number of spots in the caseload that will open in the next 28 days. This count is based on the amount of care needed for each of the patients currently being served. As the next day's patients arrive, the length of the queue is limited to this number of "soon to open slots". Patients who arrive when the queue is full (those that would not be seen in the next 28 days) are rejected from the system of care.

Purchased Care Systems The simulation sends all excess demand to the purchased care system. The simulation assumes that purchased care networks readily accept any patients sent to them. This is unrealistic, but combined with other assumptions, it allows us to make a useful measurement:

[17]In the ARENA model patients actually get one seventh of an encounter per day. This helps with the simulation's book keeping. Because patients always require a whole number of encounters, this does not impact the model's behavior.

the amount of excess load a system is facing at any given time. The discussion of assumptions and limitations on page 120 explains these assumptions in more detail.

5.2.4 Daily Record Keeping

Each day, the simulation records the total number of patients who try to seek care at each site's MTF and the eventual destination of those patients (The patient's own MTF, a TH provider, or the patient's local purchased care system). The number of patients sent to each site's purchased care system is recorded independently, since these represent distinct markets which must absorb excess demand. The simulation also records the maximum length of each queue, the actual number of patients in each queue, and the provider utilization rate in each of the direct care systems.

These variables account for most of the outcome indicators of access to care described in [Aday and Andersen, 1974], each of which is discussed below.

1. *Satisfaction:* The simulation does not explicitly model satisfaction. Satisfaction is a worthwhile goal, especially in the deeply subjective space of Psychological Health. One can reasonably assume that improved continuity of care, waiting times and volume of care will positively impact satisfaction. The data to properly model satisfaction as a result of experience was not available and would likely require a much more complicated model of the system including aspects like patient/provider relationships and personal preferences.

2. *Continuity of Care:* The model assumes that patients complete their course of treatment where it began, regardless of circumstances. Ideally, each patients would recieve an evidence based course of treatment the properties of which don't change with increased system load. By holding continuity of care constant in our simulation, we can better elucidate the demand overload at each site.
   In psychological health, continuity of care is primarily a function of patient provider relationships, "Can you see the provider with whom you've already built a relationship?". In this respect, we can expect TH to improve prospects for service members who deploy or are stationed at another installation. Without TH, they typically stop seeking care, or transition to receiving care from another provider. Credibly modeling this movement would make a valuable contribution to this analysis, but doing so is impractical.
   Data on the site to site movements of providers and an adequate description of how the system currently adapts to such movements were unavailable. In addition, we do not know the effect of imminent deployments on a patient's decision to seek care[18]. Site visits have shown that individual installations and providers have significantly different practices for ensuring continuity when their patients are reassigned, deployed or separated from service.

[18] While our data can tell us the number of patients who used care and then deployed, it can not tell us how many were supposed to deploy but did not because of a psychological issue.

3. *Waiting Time:* Patients who must wait too long to see their provider suffer. When waits are too long, demand may be suppressed because of balking or blocking. In addition to typical balking behavior, patients in the military may balk because the next available appointment is after their scheduled deployment or permanent change of station.
   Waiting time is imminently measurable, and the model described here rigorously tracks the wait time between seeking care and receiving it. Since our demand function is generated at a daily granularity, wait times are reported with the same precision[19]. Day of appointment waits are ignored in this simulation because credibly modeling the micro-level operations of all individual clinics is infeasible.

4. *Volume of Care:* Psychological healthcare is seldom a one stop affair. First line treatments for PTSD and other conditions consist of series of encounters over the course of weeks. A military working group recommended, for example, providing an average of 16 encounters to a patient with PTSD [Harris et al., 2010]. This model assumes the same demand for volume of care per patient as PHRAMS does.
   Given the especially subjective nature of psychological health, clinicians may be reasonably assumed to "do the most good" with their time. This may include checking up on patients more frequently when time permits, and focusing on more severe cases at the expense of less severe ones when they are overloaded. Indeed, researchers studying the Veterans Affairs system have documented reduction in the volume of care provided to each patient when demand for care is high [Rosenheck and Fontana, 2007].
   Triage practices vary from site-to-site and provider-to-provider. Modeling this nuanced behavior would be impractical and possibly misleading. Instead, we assume that the military intends to provide a consistent quantity of care at all times. Since the amount of care provided is held constant in the simulation, it is not measured.

In the simulation, we explicitly track wait times from care-seeking to first appointment. Measuring satisfaction with a servicemember's TRICARE benefit is outside of the model's scope. Continuity and volume of care are held constant.

[19] Assumption: Day-of-appointment waits can be safely ignored. Many Operations Research models have examined day-of-appointment waits in healthcare systems, which are very sensitive to facility design and operation. As discussed above, facilities vary greatly even within services, and credibly modeling such facilities is difficult and time intensive for each site. Doing so for every site is infeasible.

6 Do Deployment Related Care Seeking Behaviors Impact Installation Care Systems?

The staffing models discussed in Section 2.3 consider demand as an aggregate over time, typically as annual demand. This assumes that so long as the quantity of care available is equal to the quantity of care needed, all will be well.

The Army's analysis of the M2 data shows that Army installations in the continental US deliver between 40% and 100% of their mental health care through the direct care system [?]. Harris' analysis of Army and Air Force data from the same source shows that the direct care system provides 90% of active duty care and smaller percentage of dependent care (38% in the Army and 29% in the Air Force). At most installations, not all demand for mental health care can be met by available direct care providers. The local cadre of purchased care providers is expected to absorb the excess demand that can not be met by the MTF.

If an installation's demand for care were constant, wait times at the MTF and the number of beneficiaries sent to the network would be predictable. But, under the widely varying arrival rates documented in Section4.5, our service system should see more variation in queue size and in the number of patients who are sent to the network. Local purchased care providers can not offer acceptable access to care if arrival rates vary too much.

Section 6.1 describes the method for comparing the installation level impacts of constant rate arrivals and the projected arrival rates. 6.2 briefly describes the impacts on a representative installation (an Army power projection platform) and 7.3 explains the implications of those results.

6.1 Experiment

This experiment determines whether the demand patterns projected in 4.5 make it harder to meet Acess to Care standards than constant rate arrivals would.

Two scenarios are compared. Only the arrival patterns differ. In both, patients can only recieve care from their local resources (direct care or purchased care). The total number of arrivals in both cases and the amount of care available are the same.

6.1.1 Scenarios Compared

Scenario 1a: Constant Arrivals to the System New patients arrive at a constant rate for the duration of the simulation. Arrivals are poisson distributed at an installation-specific rate equal to the *average* number of arrivals per day projected in Section 4.5. The amount of care available is equal to half of the expected demand. In this scenario, all patients seek care at the MTF first. If a provider is available immediately, they are seen immediately. If not, they will make an appointment as long as their expected wait is no greater than 28 days. Any patients that can not be seen by the local direct care providers is sent to the local purchased care system.

Scenario 1b: Varying Arrival Rates New patients arrive at the rates projected in 4.5. Arrivals are still poisson distributed, but each day's arrivals are calculated independently using the expected arrival rate for that installation on that day. The same amount of care is available as in scenario 1a, and patients seek care according to the same rules.

6.1.2 Dependent Variables

We compare the system's performance under each scenario by examining several metrics for each MTF:

1. The rates at which patients are referred to the network over time (and a histogram, showing the variance in referrals from month to month)
2. Provider utilization over time
3. Queue length for beneficiaries who do get access to the direct care system (a full queue is a full 28 day wait)

6.2 Results

A summary of system performance at Army Site Alpha for each scenario is presented in Figures 35 and 36.

In the first subfigure of each page, we can see that arrival rates over time are different. In scenario 1a, the rates are relatively constant, averaging 56 per day. Arrival rates fluctuate modestly over time because of the inherently random poisson arrival process. In scenario 1b, however, we can see a pronounced trend in arrivals matching the projection in Figure 24.

The surge in arrivals in scenario 1b is the result of the 12,000 returning soldiers. For about 200 days, demand stays elevated, then drops back down to a lower rate than before an another 10,000+ serivce members left site Alpha for a deployment.

In both scenarios, we see that only a subset of the arrivals is accepted into the local direct care queue (dark green). These rates fluctuation randomly, but are essentially identical. In both cases, all direct care providers are busy 100% of the time (third subfigure), and all new arrivals who did get care in the Direct Care system waited the full 28 days because the queue (representing the appointment book) was always full (fourth subfigure).

Looking at the second subfigure of the scenario 1a summary, we can see that a relatively predictable number of patients (about 900) are sent to the purchased care network each month. But, when the system sees realistic arrival patterns in scenario 1b, the number of monthly referrals to the network varies widely. During lulls in demand, about 650 patients per month are sent to the network. During periods of peak demand, more than twice that number are sent to the network for care. While demand peaks vary in their frequency and their duration (as we can see by looking at the results from other sites), we can see that they are rarely followed by a return to business as usual. Rather, a short surge is followed by a prolonged period of higher than average arrival rates.

6.3 Discussion

Local purchased care providers may not have the capacity to absorb surges in new patients. We can see that the difference in demand from a slow month to a high-demand month is enough double the demands put on the local purchased care system.

Many installations are located in sparsely populated areas. In a small community any mental health practice that could handle the influx of patients during a demand surge would be underutilized when that demand fell off. Relying on GAO and TMA surveys, [Tanielian and Jaycox, 2008] suggests that providers are sensitive to TRICARE's low reimbursement rates for outpatient mental health care. If providers are sensitive to low reimbursement rates, it is hard to believe they would plan their clinic around handling such surges - especially when thir practice is not responsible for meeting MHS's access-to-care standards. The same study found that purchased care provider shortages are most pronounced in geographically remote areas.

Using conventional benchmarks, we can see the amount of *additional* care required to support an installation like Army Site Alpha during its peak months is enough to completely displace the care for 92,000 other local residents. A quick calculation shows that one new patient per day is roughly enough to keep one provider busy full time:

$WeeklyEncountersNeeded = PatientsPerDay * 7 * EncountersPerPatient$

$= 1\frac{patient}{day} * 7\frac{days}{week} * 5.0871\frac{encounters}{patient} \approx 35\frac{encounters}{week}$

PHRAMS and other models estimate provider productivity to be approximately 35 encounters per week. The US Department of Health and Human Services considers a location to have a shortage of mental health providers if its ratio of patients to care providers is greater than 4,600:1. At that rate, a differential of 20 mental health providers is equivalent to the entire capacity needed to serve $20 * 4,600 = 92,000$ people. Again, because the utilization calculations rely on historical extrapolation, this is a conservative estimate.

Demand overloads like this can hurt service members and their families. Because of the priorities set forth in federal regulations [?], the excess patients pushed to the network will be almost certainly be the spouses and children of service members. If network providers can't accept them, they will not get the care they need. In addition, if demand from active duty service members outpaces the capacity of the direct care system, they too may be in trouble. Service members are not generally

Figure 35: Constant Demand Pattern: Army Site Alpha

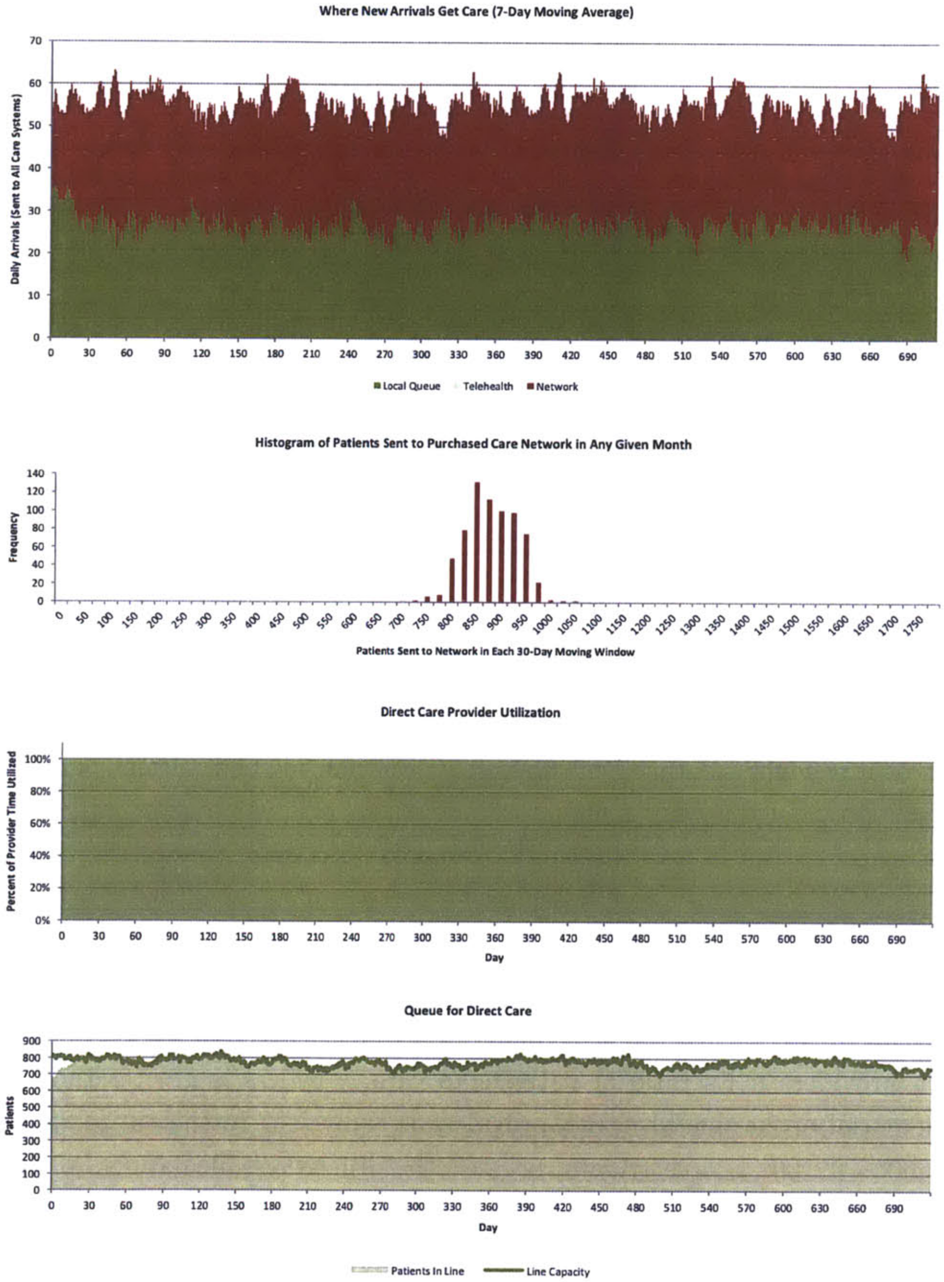
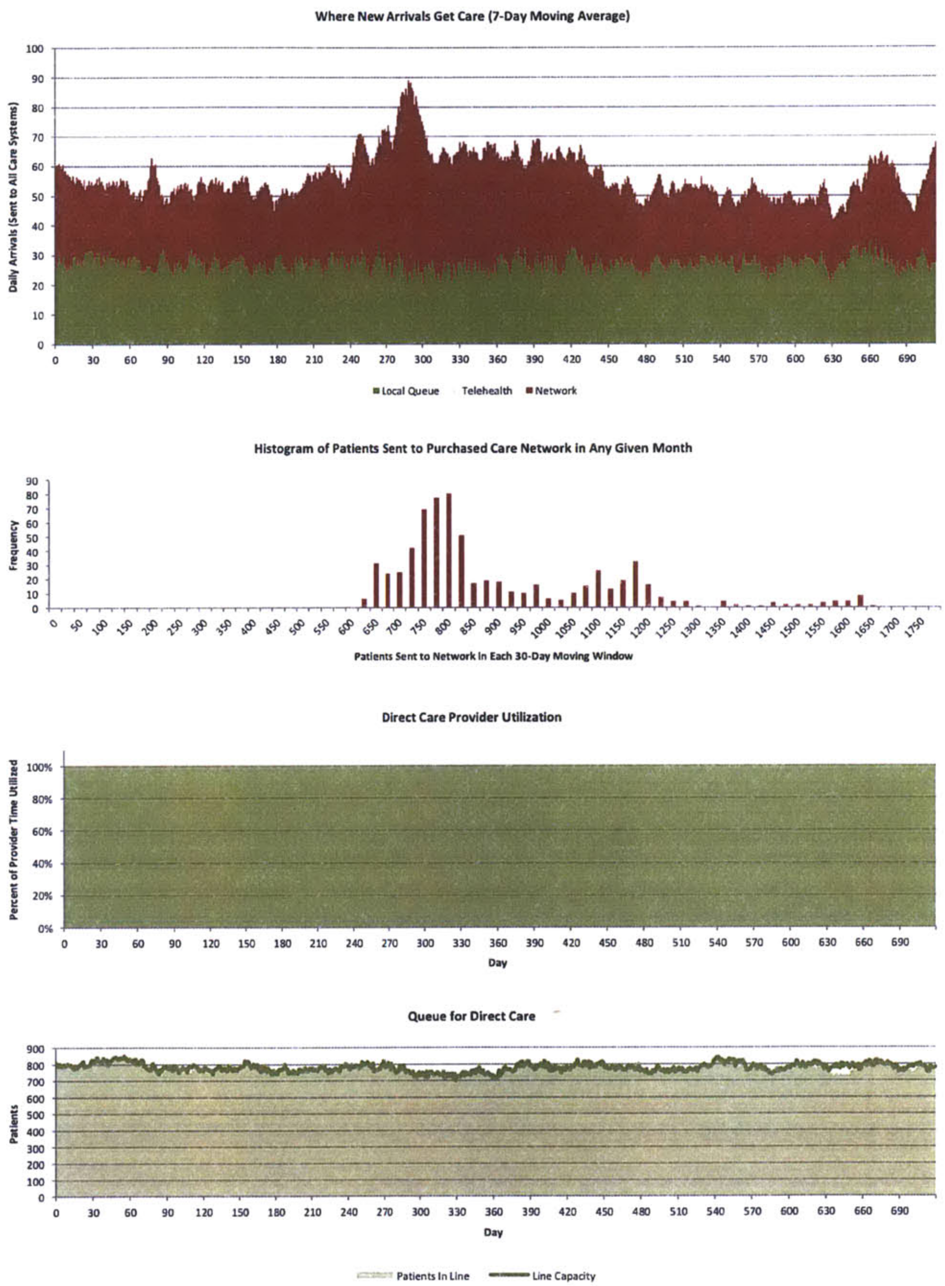


Figure 36: Realistic Demand Patterns: Army Site Alpha



permitted to seek off base mental health care because command would have no visibility into their deployability or fitness for duty (which typically includes carrying a weapon)[20].

While there have been no assertions that service members are denied care outright, a demand overload can manifest itself in different ways. First, providers may accept all new arrivals into their caseload. Oversized caseloads were a complaint among providers we interviewed. With a very large caseload, providers either sacrifice quality of care by seeing their patients less often or work overtime which can lead to burnout. Not incidentally, the DOD is concerned about turnover rates of their providers[21]. Second, knowing that there is a long wait, service members may not voluntarily seek care they otherwise would have. With frequent moves, deployments and training excercises, service members may defer future treatment that conflicts with military obligations.

[20]Some sites do allow service members to seek care off post, but the proportion of ADSM care provided in the purchased care sector is usually insignificant

[21]The DOD's proposed "Psychological Health Imperatives Dashboard" explicitly measures turnover rates. High turnover rates in some medical specialties pose challenges to knowledge management and to "continuity of work and responsibility" [Casscells, 2007].

7 Can Telehealth Mitigate Access-to-Care Problems?

Part of telehealth's appeal is load balancing, using excess capacity at one site to meet excess demand at another.

Instead of accepting only local ADSMs and family members, providers will soon have the additional option of treating distant patients. When this new option emerges, providers may prioritize distant service members over local families in accordance with federal regulations. Load balancing arrangements could take several forms including:

1. No load balancing at all.
2. Sites only support one another when all local demand is exhuated. Exhausting all demand is very unlikely.
3. Sites only support one another when all local service member demand is exhausted. Exhausting all service member demand is still unlikely.
4. Sites draw on dedicated telehealth providers (the "TH cell") to support them during surges. A few technological and policy changes could make this a reality.

This section compares each of the arrangements above, with the exception of number two. Sites in our simulation never exhaust all local demand, so the results from this would be equivalent to number one.

Each of these scenarios is described in more depth, and the grounds of their comparison are presented in Section 7.1. A straightforward description of results is presented in Section 7.2, and a full discussion of their implications follows in Section 7.3.

7.1 Experiment

To better understand the effects of telehealth network on access to care, we compare several real and hypothetical telehealth architectures.

7.1.1 Scenarios Compared

In each of the scenarios, we use the same arrival pattern as we did in scenario 1b. Care capacity at each site is equal to 50% of the average demand put on the system over the course of the simulation.

Scenario 2a: The Status Quo Equivalent to scenario 1b, this is the case where sites are left to treat their own populations.

The military does offer telehealth today, but only through an inefficient mechanism. Fragmented scheduling systems make it very difficult to distribute patients to distant providers on an as-needed basis. Instead, TH cells today offer care in weekly allotments. A TH cell will offer a fixed number of weekly appointments to each distant site. From the perspective of load balancing, this is equivalent to increasing the local cadre of providers, since this care can't be readily offered to another site in need.

Scenario 2b: Sites Assist Each Other At times, installations rely on one another's providers for support. During the returns of Brigade Combat Teams (approximately 4,000 soldiers), some sites will request a few days worth of time from a handful of providers to help perform post deployment screenings. This approach, more of an axe than a scalpel, helps meet the immediate needs over the first few days. But, patients who are diagnosed or referred for further evaluation usually go on to see a local provider (a lapse in continuity). Distant support providers also aren't available over the coming monthst to help handle the 90-150 day swell in new patient arrivals.

In this scenario, we analyze access to care when sites can rely on one another's providers for *on-demand* support (not just one time sharing of blocks of provider time). Service members take universal priority, meaning that a distant service member will be served by an MTF before it accepts local family members into its caseload. There is no TH cell available, so the group of six sites can use only their own combined resources.

Scenario 2c: Unlimited TH Cell Support for ADSMs This case simulates the current "Hub & Spoke" architecture, but with an unlimited capacity and a flexible scheduling system. Instead of offering each site a finite number of encounters per week, the hub offers on demand care for any service member who needs it. Telehealth care in this scenario is provided by a TH Cell (similar to the ones operated by the Army's regional medical commands). This cell does not have any local demands. Because the TH cell can take on every single excess patient, MTFs in this scenario are never asked to take on any demand from distant sites.

Scenario 2d: Finite TH Cell Support for ADSMs Last, we examine a more realistic version of 2c. We offer TH cell support to sites on the network which need it. But, we only offer a limited capacity. The TH cell capaicy is set at a level low enough that all TH providers stay utilized at all times[22]. If more patients seek care on any given day than can be accepted into newly opening TH cell slots, a random selection of the patients seeking care are accepted, and the rest referred to purchased care. Because, probabalistically, the same proportion of patients from each site are accepted, the TH cell provides more capacity to sites facing the most acute shortage of providers.

[22] Figure Figure 42 on page 106 shows that 600 spots in the caseload are almost always occupied. Suspecting that a queue would hold enough of a backlog to keep the system full, that state of affairs was simulated. The suspicion was confirmed, and 2d presents that run's results.

7.1.2 Dependent Variables

We compare the scenarios using the same variables as the last experiment. In addition, we analyze the TH cell.

7.2 Results

Scenario 2a: The Status Quo Examining Site Charlie's summary for scenario 2a (see Figure 37), we can see that it gracefully handled its first surge because it was preceeded by a period of unusually low demand. By the time the surge arrived at about day 360, demand at Site Charlie had dropped so low that it had completely served the entire backlog of ADSMs and begun serving family members as they arrived instead of referring them to the network.

By the time the second and third surges rolled around, Site Charlies direct care system was already filled with ADSMs, and all excess demand was pushed out to the network. The histogram in the second subfigure of Figure 37 shows the effect: the local purchased care system was saddled with a doubling to tripling in the rate of new arrivals.

Scenario 2b: Sites Assist Each Other Acceptance to the local care system at Site Charlie, shown in the first subfigures in Figures 37 and 38, differs. Figure 39 shows this modest difference in better detail. Starting near day 270, when demand from ADSMs would otherwise have fallen low enough for the system to begin admitting family members – which it did in without telehealth (scenario 2a) – the system in scenario 2b began accepting excess service members from other locations.

Not only did this have the effect of helping other sites (Site Alpha was in the middle of a major surge, for example), but it also prevented Site Charlie from sending an especially small number of patients to the network. The histograms in Figures 37 and 38 show that when Site Chalie helps out other sites, it never sends less than 150 patients per month to the network. The effect in this simulation is small because Site Charlie had barely any capacity to spare over the course of almost 2 years, but it does demonstrate the potential for site to site load balancing.

Unfortuntately, as the time series in the first subfigure and the histogram in the second subfigure of Figure 38, other sites did not have excess capacity with which to help Site Charlie absorb its later surges.

Scenario 2c: Unlimited TH Cell Support for ADSMs When there is unlimited telehealth support available for Active Duty Service Members, a consistent number of patients (all family members) are referred to the network and surges do not cause large variation in excess patient load (measured in referrals to the network).

Figure 37: The Status Quo: Army Site Charlie

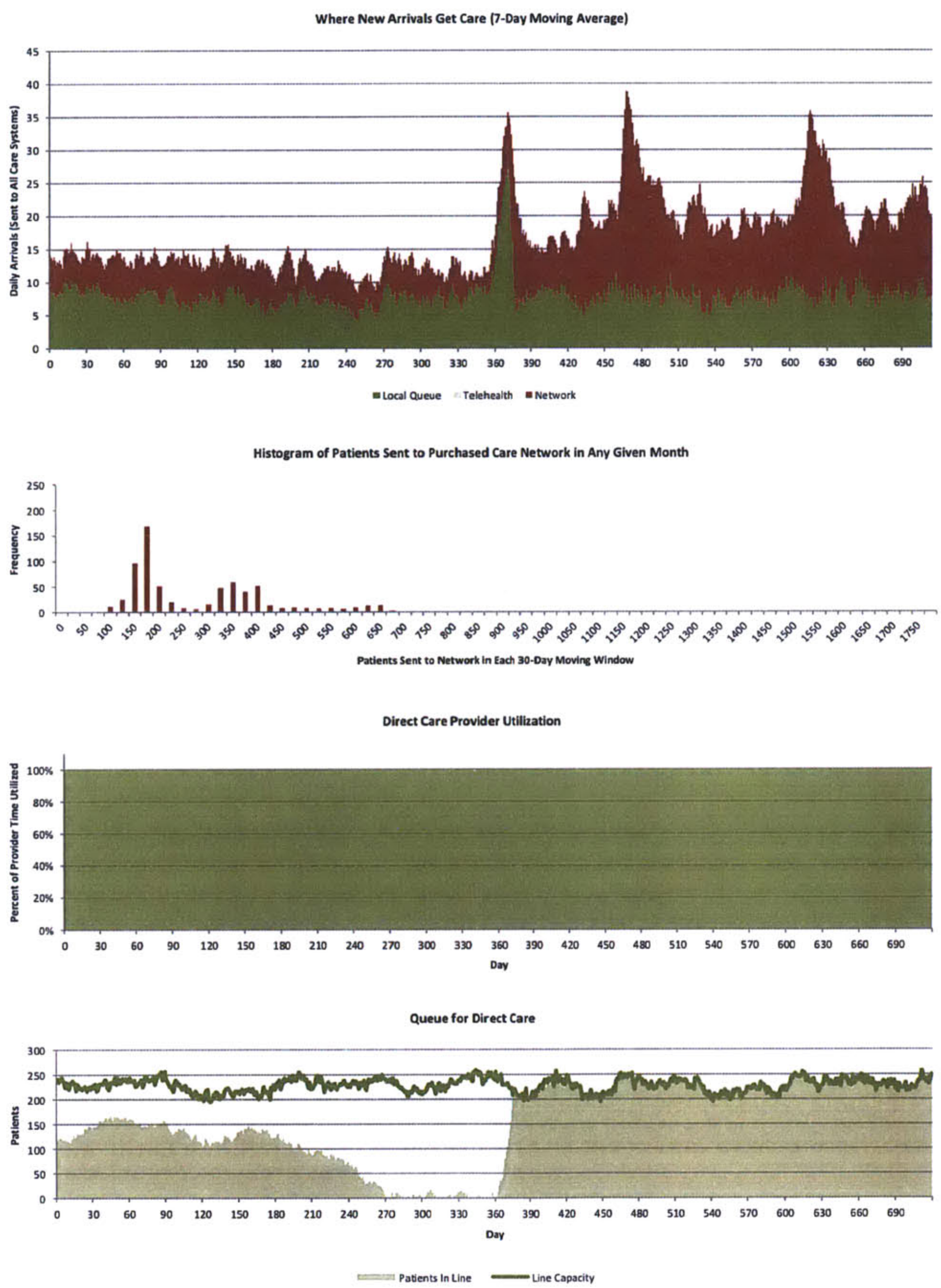


Figure 38: Sites Assisting Each Other: Army Site Charlie

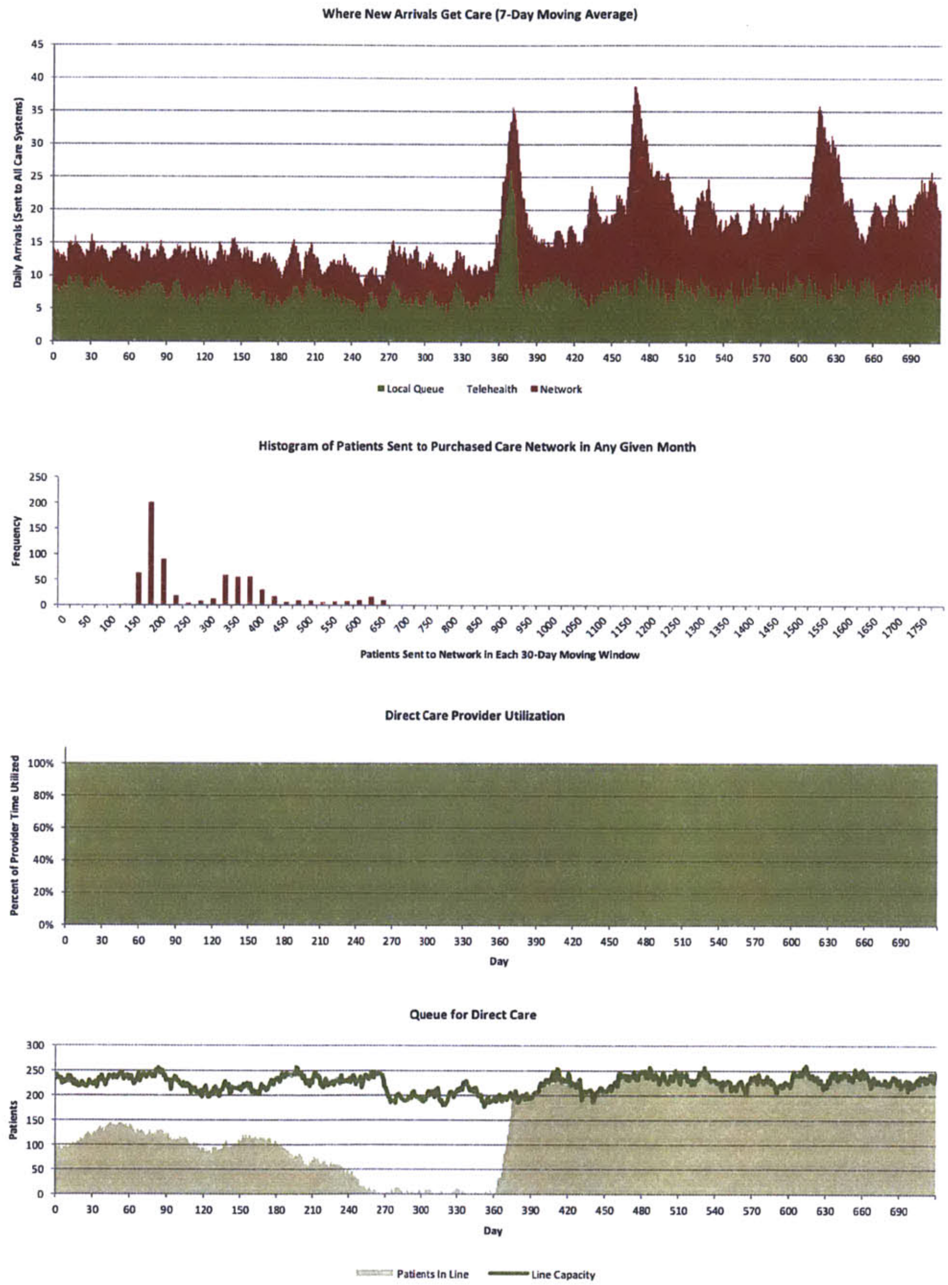


Figure 39: Comparison of Scenarios 2a and 2b: Acceptance to Direct Care at Site Charlie

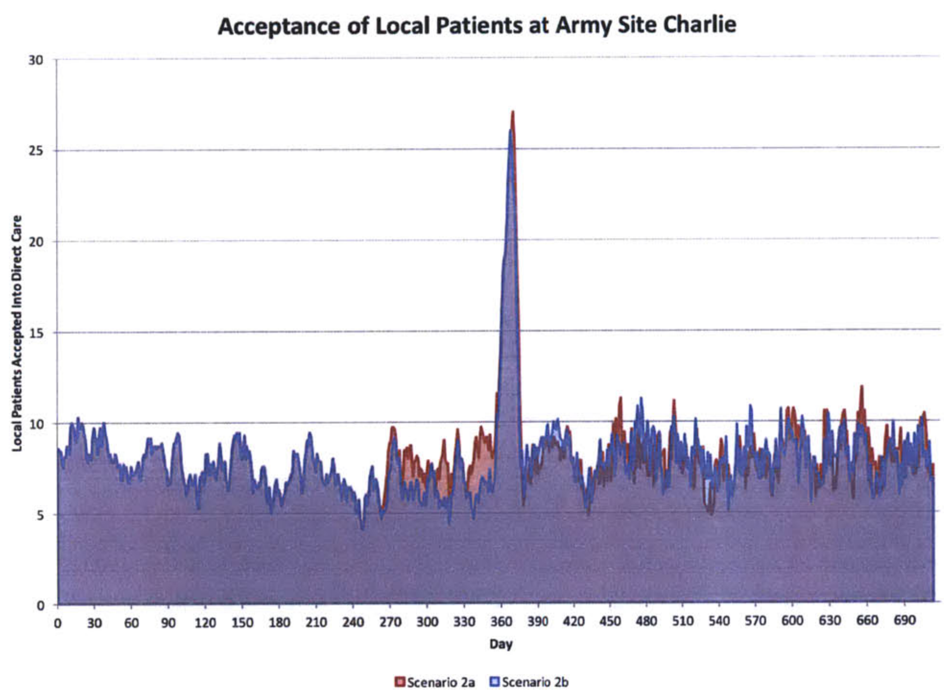


Figure 40 shows that the surge which would almost certainly have overwhelmed Site Alpha's purchased care system gets picked up by the unlimited TH cell providers standing by. All service members at Site Alpha stay in direct care, either locally or via telehealth. Because they have the first right of refusal for service members and their families, Site Alpha's direct care providers remain completely occupied for the duration of the simulation.

Looking at the queue for direct care, we can see that it is nearly always full. This indicates that patients seen at Site Alpha usually wait the full 28 days for an appointment. Family members are only accepted when no service members are waiting. Since there's always a line, we know that all family members in this simulation are sent to the network.

In this scenario, the local care system is not asked to take on the fluctuating arrivals from ADSMs. Since family member arrivals are relatively constant over time, this leaves individual MTFs facing relatively constant levels of excess demand (compare the second subfigures of Figures 36 and 40).

The story is much the same at Site Charlie (Figure 41) and in all the other sites which see spikes in service member demand. At Site Charlie, the surges it was previously left to handle on its own are picked up by the TH network, and the local purchased care system sees almost no variance in demand. With unlimited TH available on the network, no other sites seek help from Site Charlie. Note that in this case we see the small tail on the low end of the histogram (months with less than 150 excess patients) because Site Charlie begins accepting family members into its direct care clinics between days 270 and 360.

The hypothetical TH Cell in this scenario handles every site's spikes in demand. New arrivals to the TH system fluctuate significantly (subfigures 1 and 2 of 42), but the utilization of providers, shown in the third subfigure, does not vary as drastically over the span of weeks. In this scenario, all arrivals to the hypothetical TH cell enter care immediately, so there is no queue and no queue statistics are displayed. One can reasonable expect that adding a well managed queue to the TH system would reduce variation in utilization further.

Scenario 2d: Finite TH Cell Support for ADSMs Of course, it's not realistic to expect an unlimited amount of providers to be available on demand. To examine the likely behavior of a system which offers some, but not all of the desired capacity, we set a limit on the number of providers in the TH cell. A quick appraisal of the load put on the TH system in scenario 2c shows that 600 caseload spots should be utilized 100% of the time. In our model, this equates to 600 encounters per week.

What happens at the TH Cell? The results for 2d are presented in 43. There, we can see that the TH cell sees a smaller variation in new arrivals. When the TH cell's schedule is filled for the next 28 days, patients it cannot serve are sent to the network to seek care. The queue for care fills up by day 280 when demand for TH finally outstrips the capacity of the TH cell to accomodate new patients. The small light blue area in the first subfigure (from day 280 to day 360) shows the small

Figure 40: Unlimited TH Support: Army Site Alpha

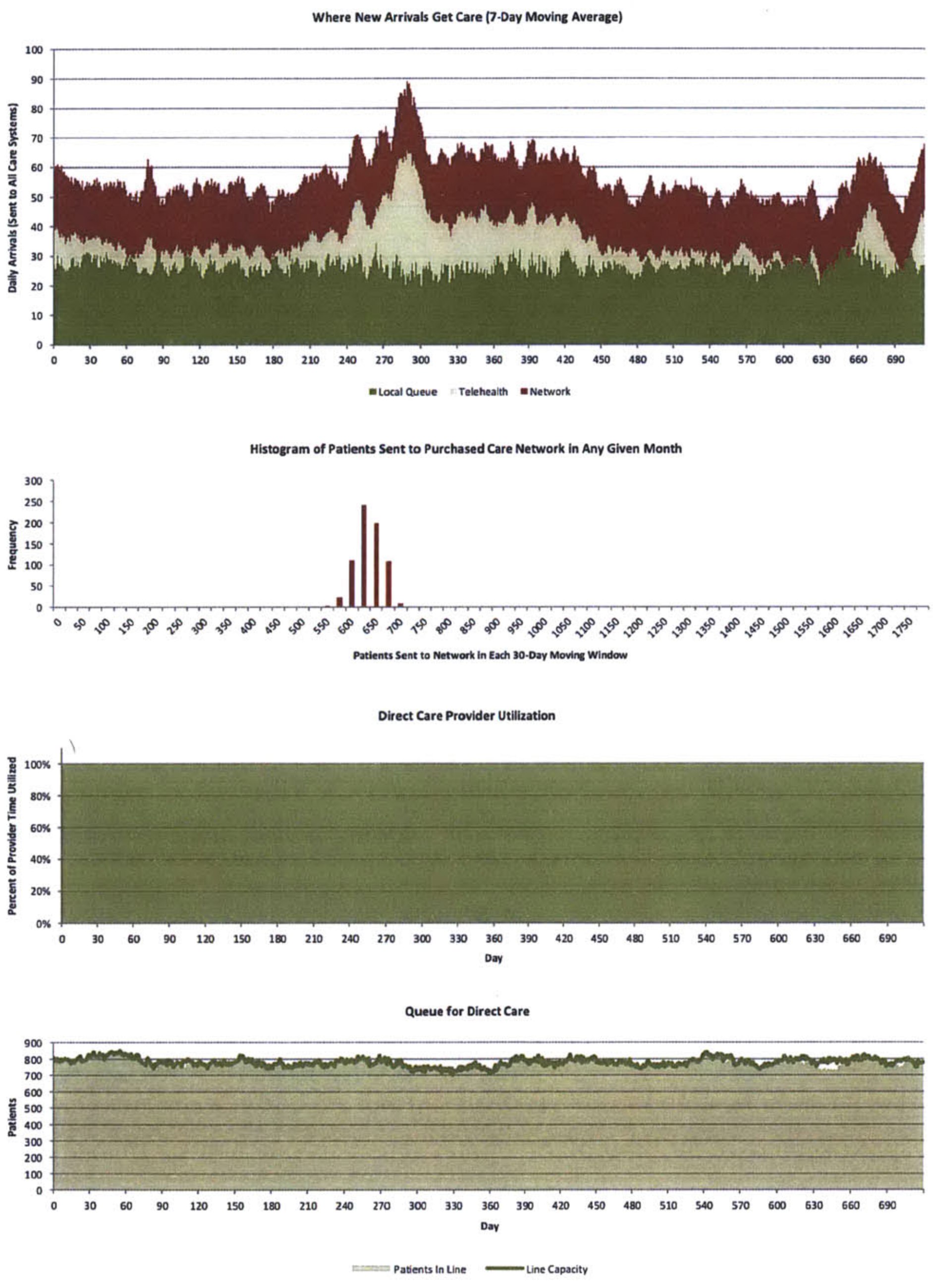


Figure 41: Unlimited TH Support: Army Site Charlie

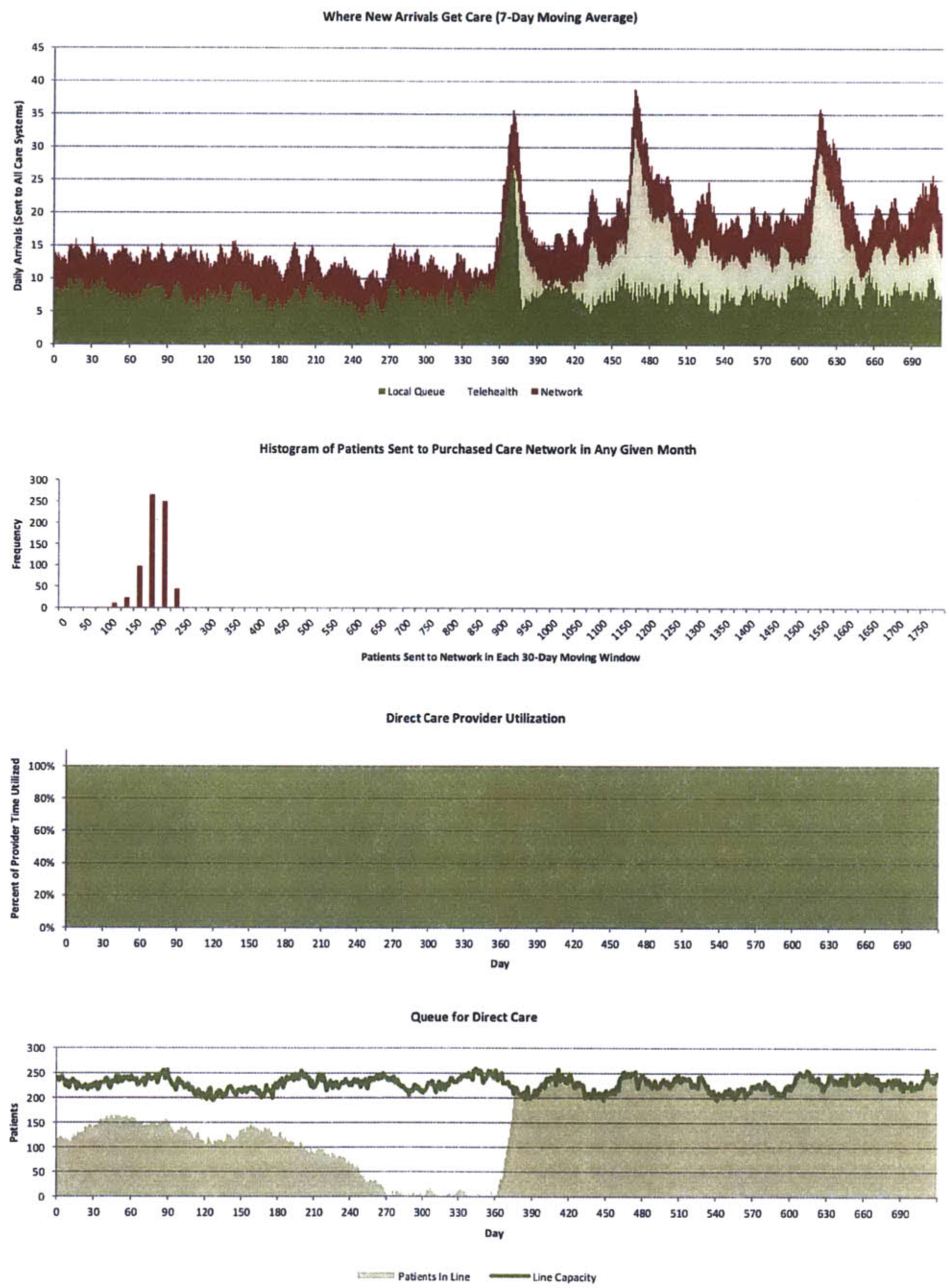
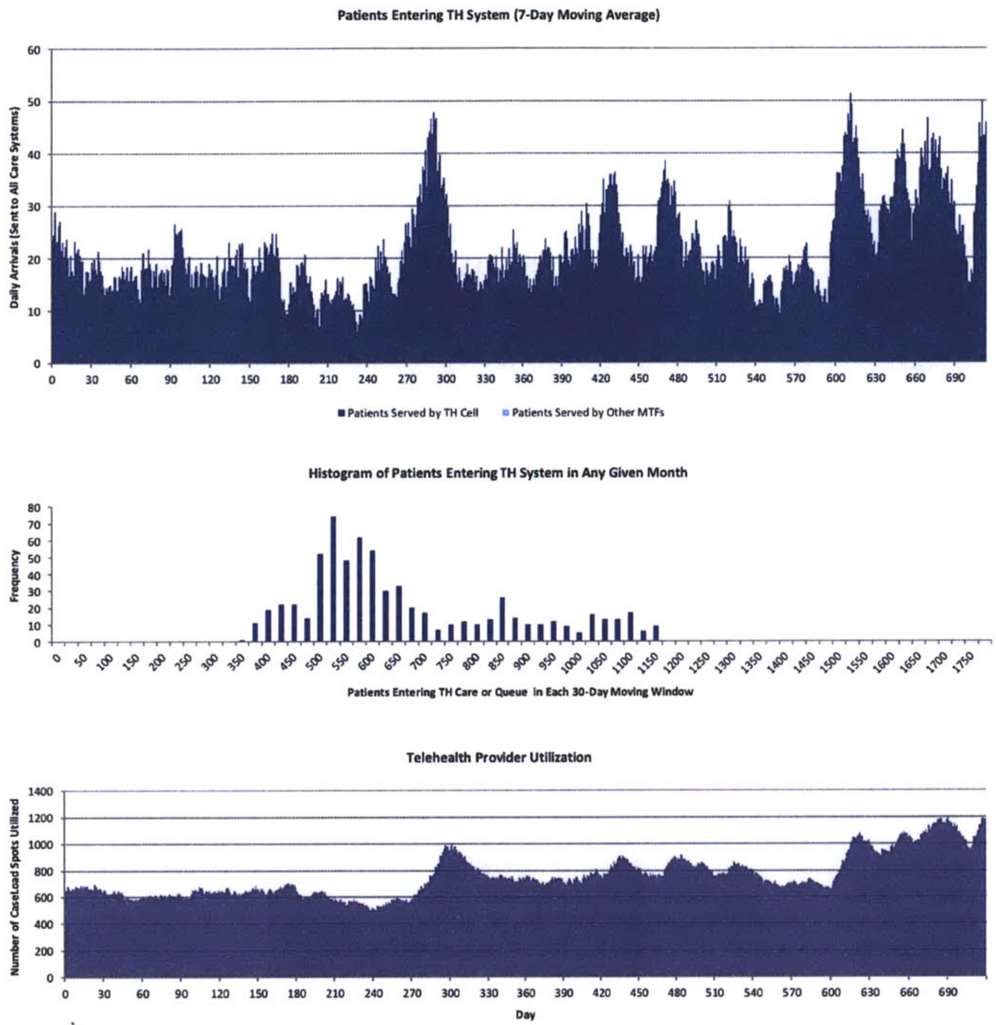


Figure 42: Unlimited TH Support: Load Placed on TH Cell



amount of capacity that Site Charlie was able to absorb during its lull. Once the TH cell's queue is filled and Site Charlie stops being able to accept new patients, not all arrivals can be accomodated. Those patients are sent to their local PC systems.

As we expected, all 600 spots in the case load remain fully utilized for the duration of the simulation. When the TH cell's capacity is close to the average demand (the first 270 days of the simulation), patients see waits of much less than 28 days because the schedule isn't filled to the brim. After that point, the TH care queue is full, and nearly all patients arrivals wait the full 28 days to see a provider.

What does this mean for the sites on the TH network? Other sites on the network can no longer rely on the TH cell to absorb 100% of excess ADSM demand. But, the TH cell does have new spots in its schedule open up each day as patients who are currently in the caseload finish their treatment. Those n spots are given to the first n patients that ask the TH Cell for care. Since patients in this model arrive in random order, this results in an equal probability that any given patient sent to the TH cell by his or her MTF will get TH care. Equivalently, this means an equal probability that the patient would be rejected, and left to get care in their local purchased care network.

This implies that at any given time, all sites will see an equivalent proportion of their excess patients accepted into the TH care system. During surges, when more excess patients arrive, sites will get more support. When they have compartatively fewer excess patients, the TH cell will provide less support to that installation.

This behavior is evident in the summaries of each site, shown in Figures 44 through 49. When the TH cell has ample capacity in its queue or in its caseload to accept patients, 100% of excess ADSMs get TH care. We can see that in the initial surge at Site Alpha (Figure 44) and during the early months of the simulation for Site Delta (Figure 47). When the queue is full, the TH cell still offers substantial support, as in the later months at Site Delta and the second and third surges at Site Charlie. All direct care providers remain fully utilized[23].

Most importantly, every MTF sees a drastic reduction in excess demand variation compared to scenarios 2a and 2b which approximate the current state of affairs. The next section explains the imporant implications of reducing this variation.

7.3 Discussion

Scenarios 2a (no site to site support) and 2b (site-to-site support when sites have served their local ADSM demand) make it clear that local installations cannot be expected to handle wild variations in

[23]The small dip in utilization for Site Charlie is an artifact of the model's design. Utilization must fall below 100% for families to be admitted. In the real world, we can expect a site with such low demand to begin scheduling family members for visits when the backlog of ADSMs is very short.

Figure 43: Realistic Level of TH Support: TH Cell Summary

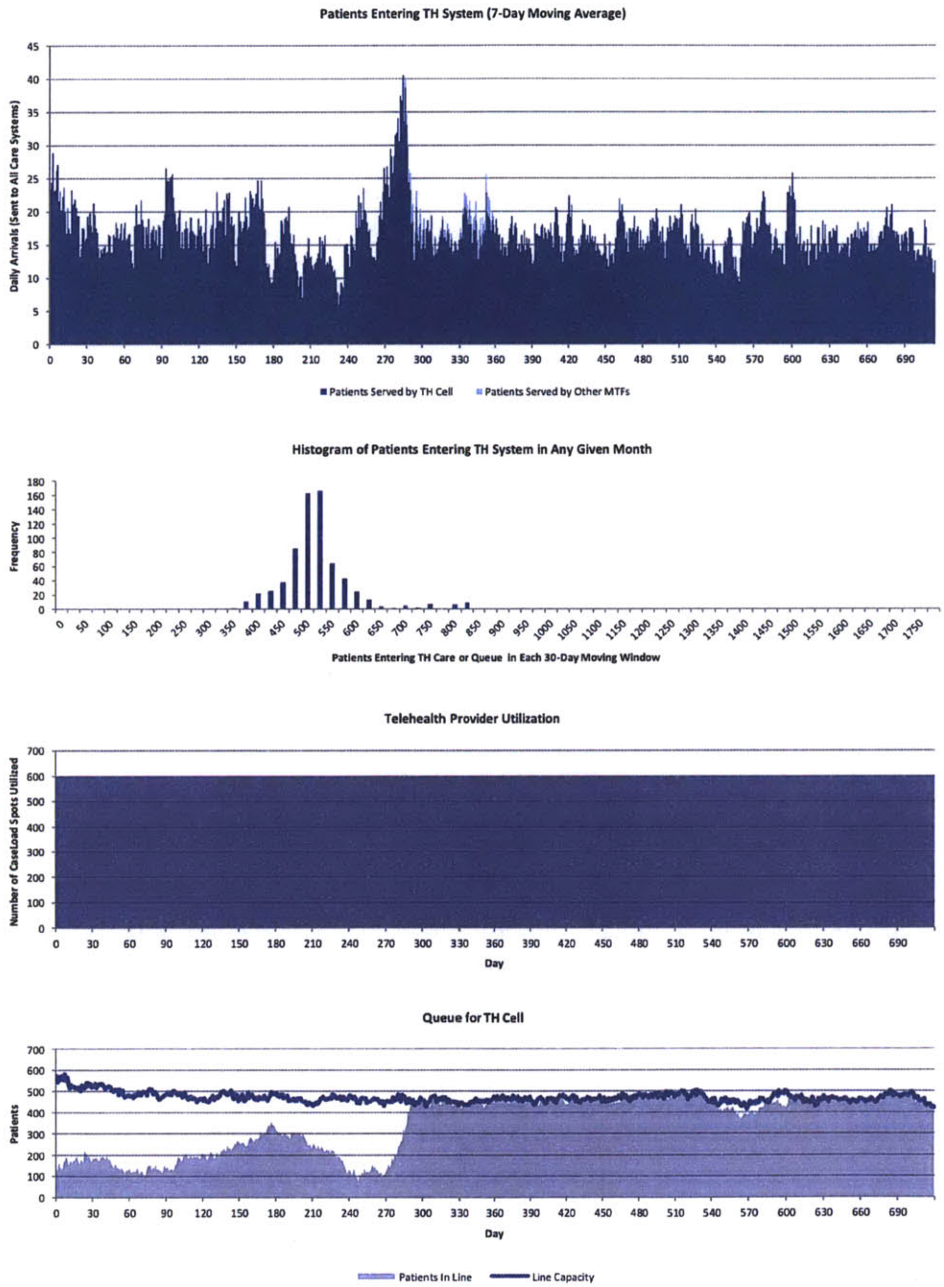


Figure 44: Realistic Level of TH Support: Site Alpha

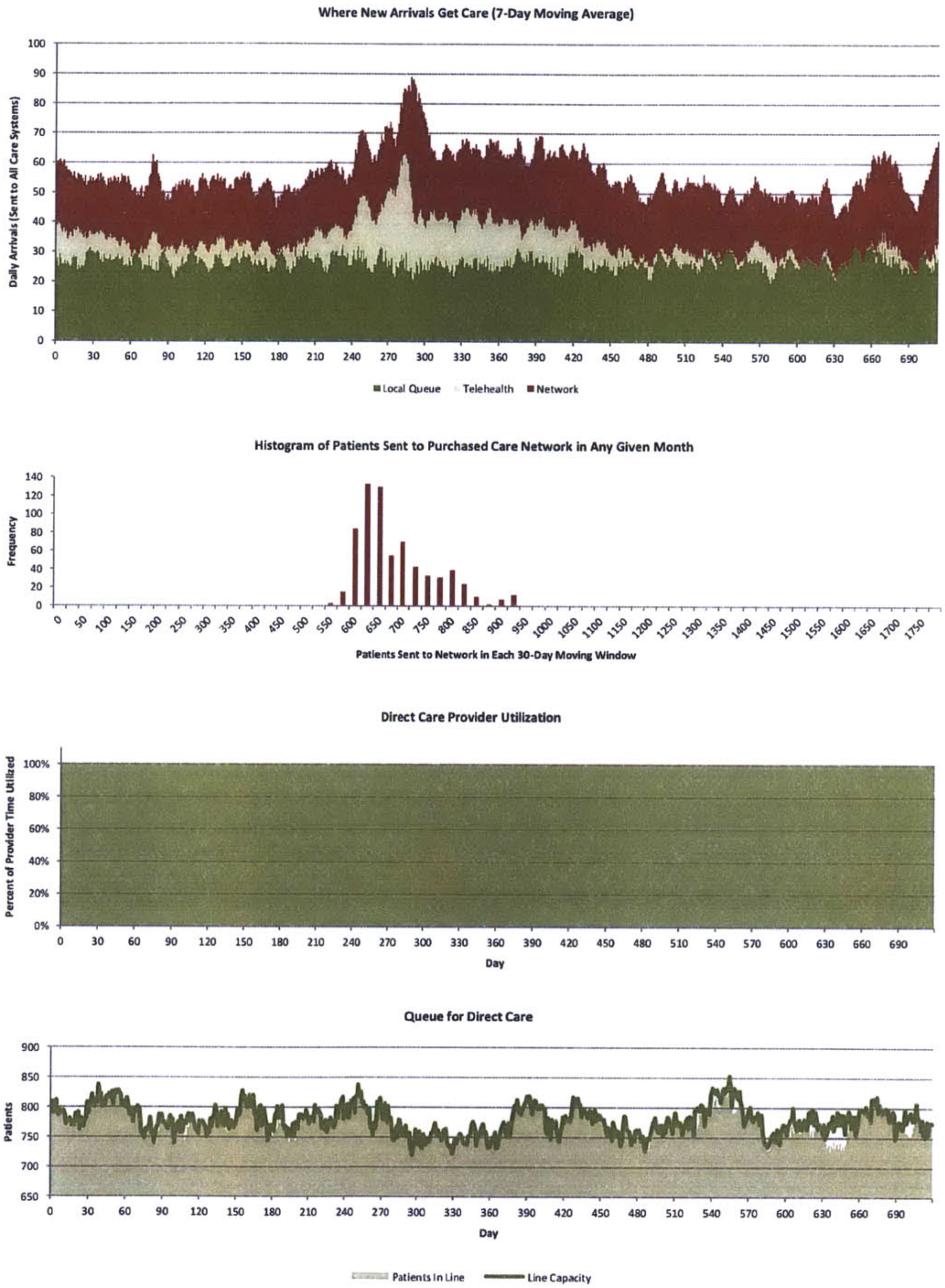


Figure 45: Realistic Level of TH Support: Site Bravo

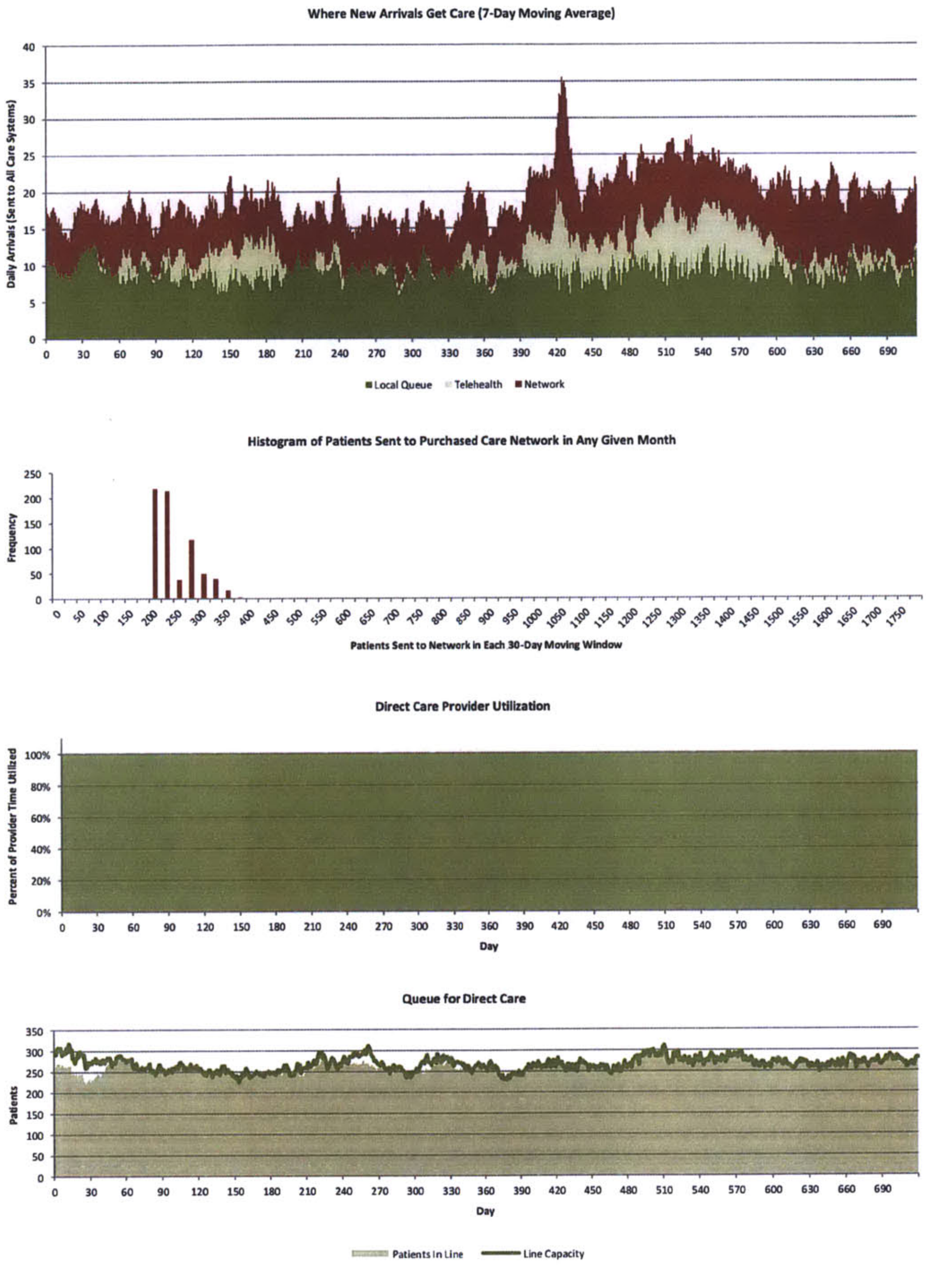


Figure 46: Realistic Level of TH Support: Site Charlie

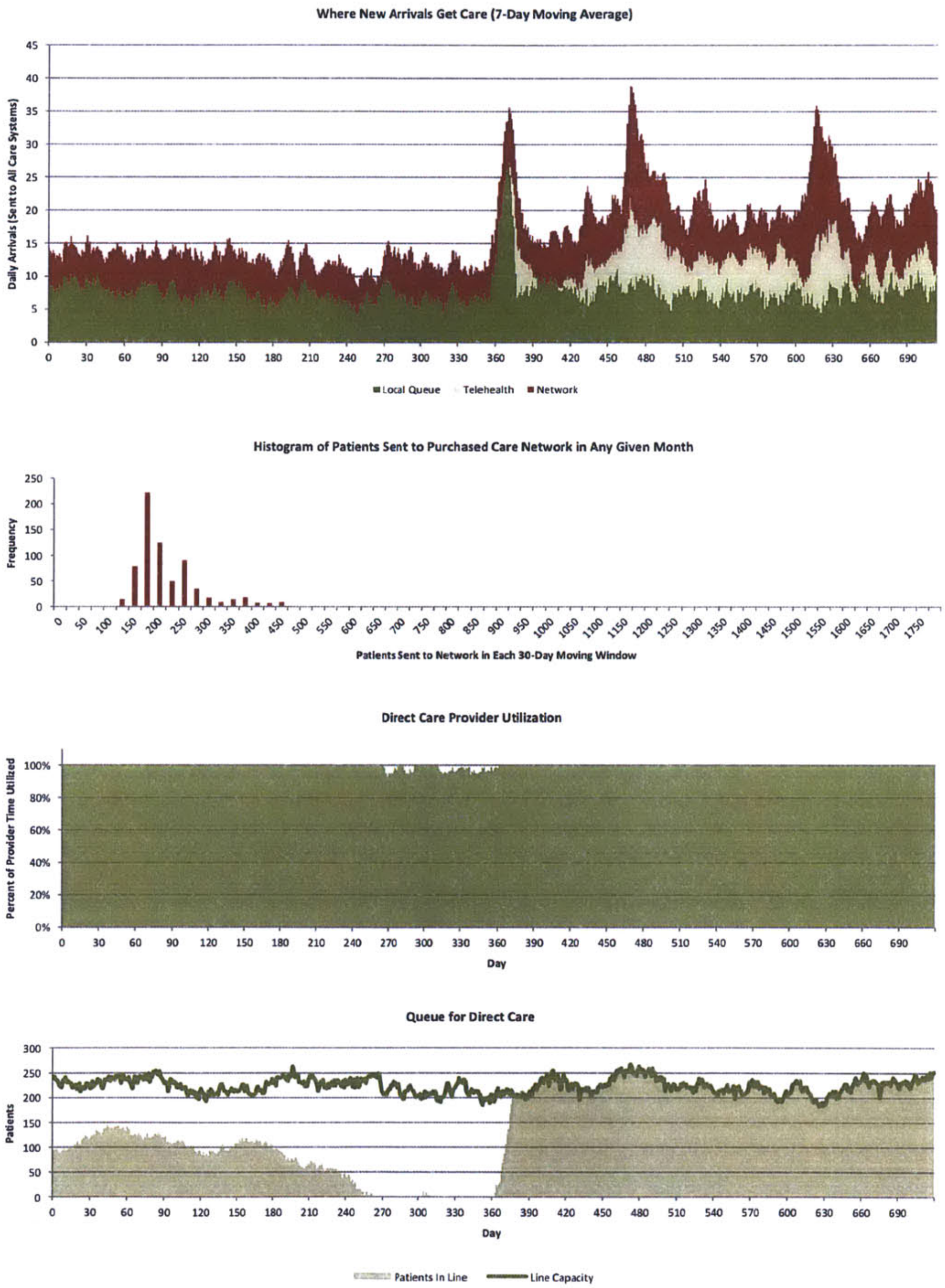


Figure 47: Realistic Level of TH Support: Site Delta

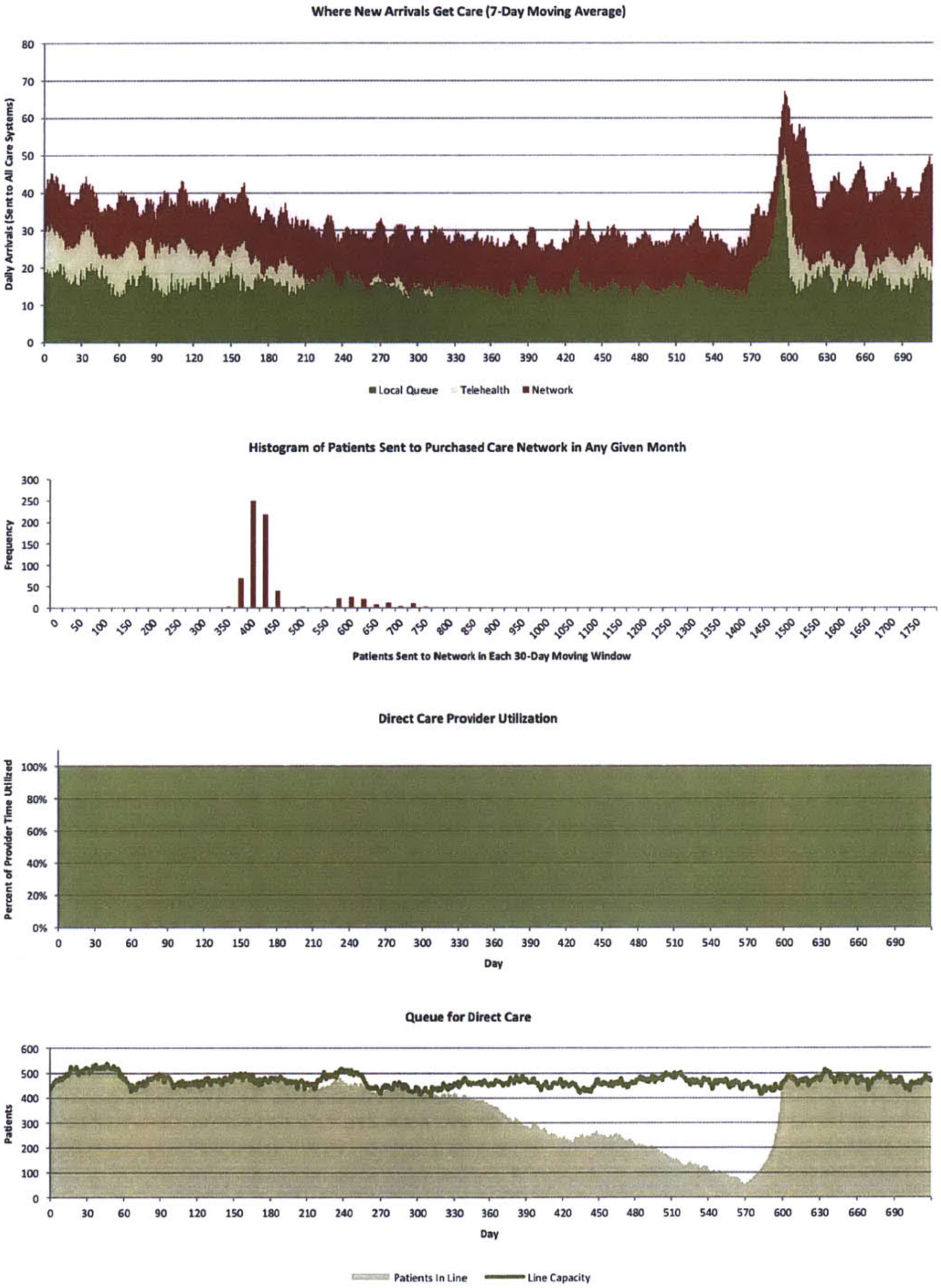


Figure 48: Realistic Level of TH Support: Site Echo

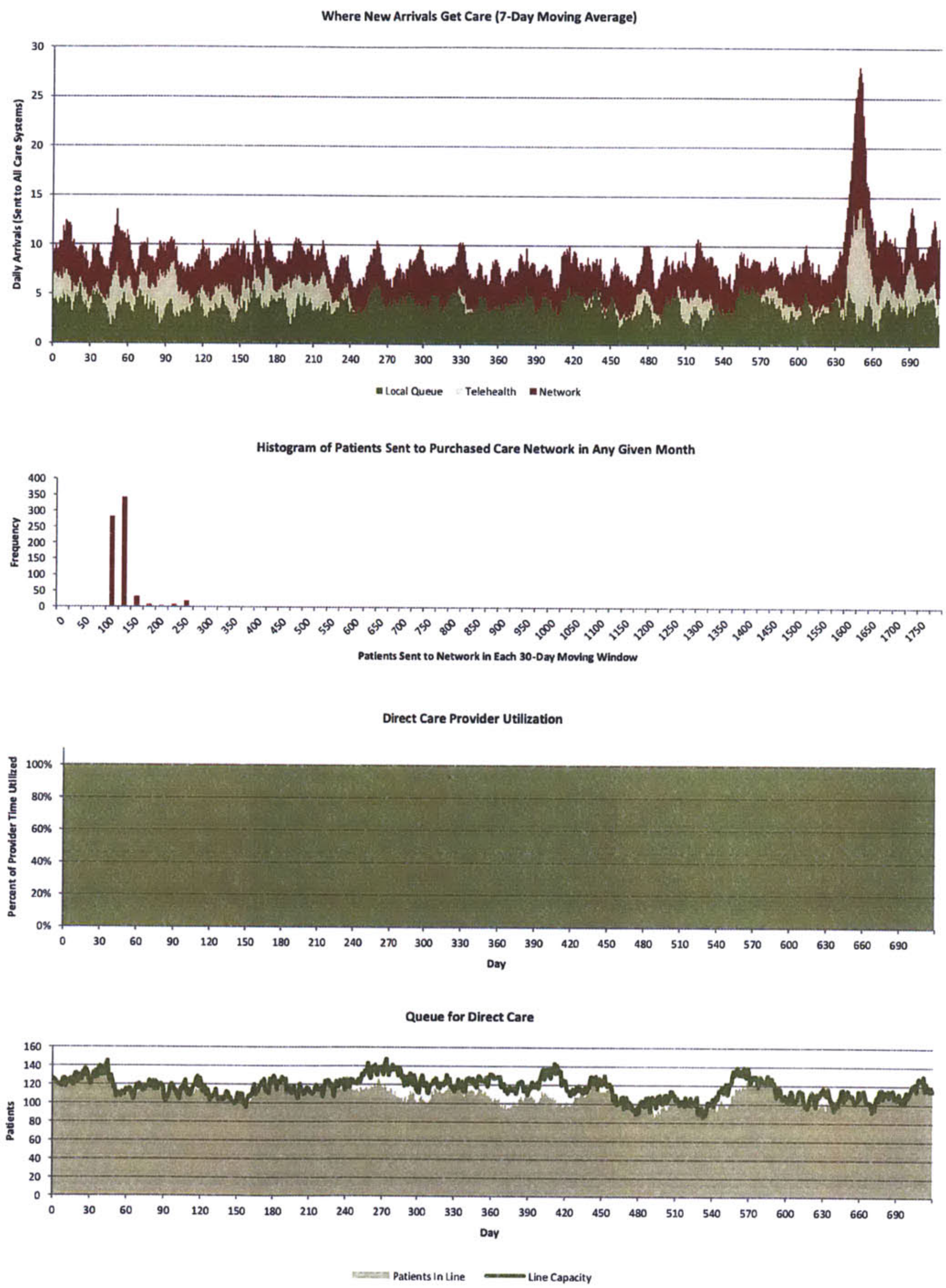
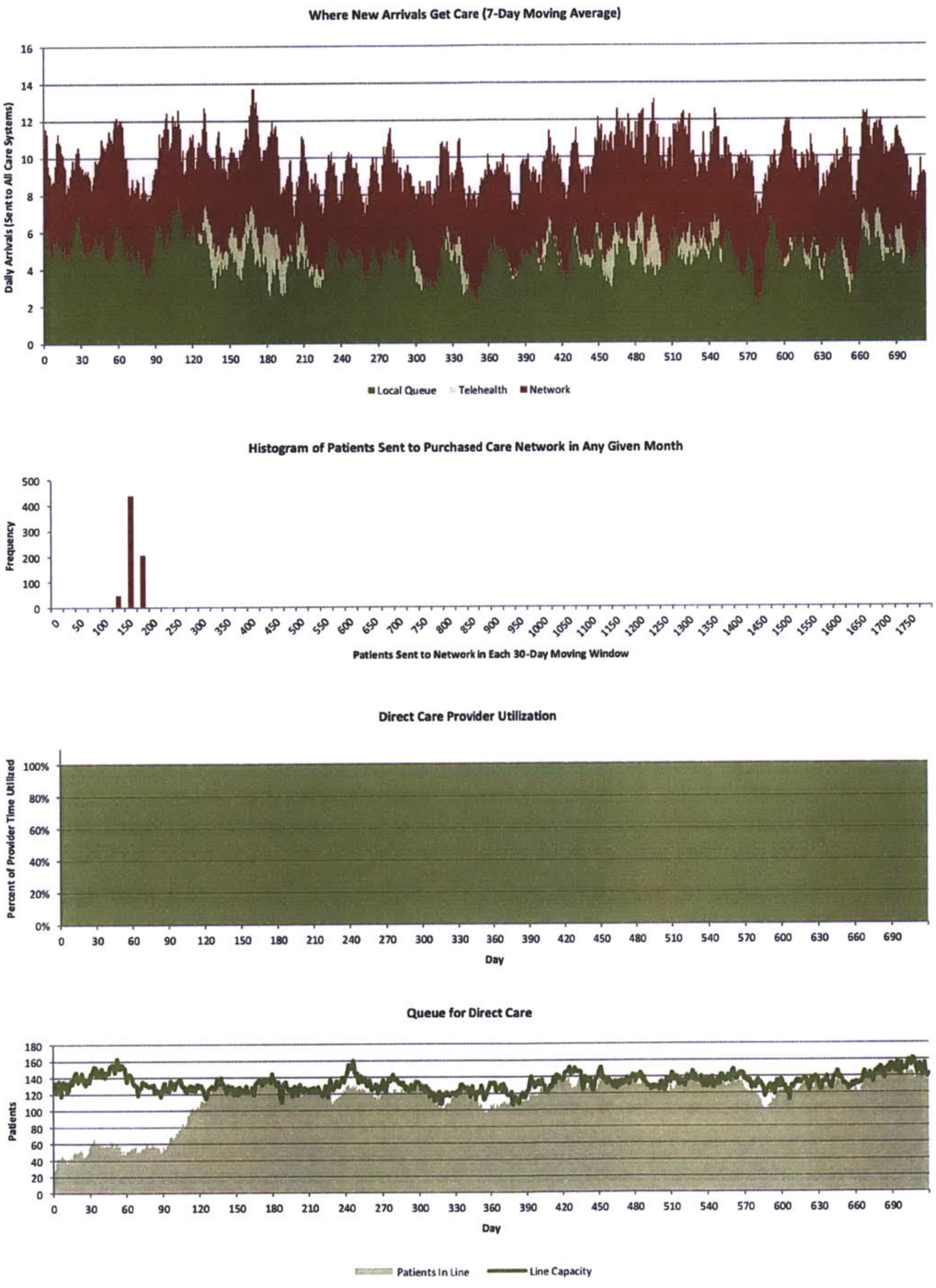


Figure 49: Realistic Level of TH Support: Site Foxtrot



demand by themselves and that they probably don't have the local capacity to help one another[24]. Scenarios 2c and 2d show us that telehealth is one promising way to reduce variation in excess demand. They also show us that TH throughput will matter, and need for it will vary enormously from installation to installation and from month to month. Lastly, all four scenarios show us that incomplete metrics and information can leave installations and the MHS blind to major access problems.

7.3.1 A constant volume of care is not appropriate

Most installations offer a constant volume of care. Granted, providers are flexible in their schedules. They can pack their training into the slow months and offer more patient time when demand is higher. Providers can also be saddled with over time when need be. But such small fluctuations in capacity can not claim to absorb 100% of the change in patient demand.

Over time, installations may hire more staff, but this is not automatic, and it only makes sense looking at long term trends. The time elapsed between deciding to hire a new provider and that provider's first encounter with a patient can be a full year, so installations can't quickly ramp up when they need additional capacity.

When an installation has varying demand and steady supply, something must happen to the excess demand. In the model, all excess demand is automatically pushed to the network. In reality though, an installation may – conciously or unconciously – use several tactics to deal with demand. Sometimes, families will be displaced from direct care to purchased care, making room for service members. When the direct care network is saturated with ADSM demand, most installations don't refer their service members off-post for care. Instead, they make room for them in the case load somehow.

Installations can "expand" their local caseload in a few ways. They can offer group sessions instead of individual therapy. They can increase the patient to provider ratio, known as the case load, so that each provider sees more patients. When this happens, either patients are seen less frequently, they recieve less encounters each, or providers end up working overtime. Given that provider turnover is a concern in the military, placing the burden of increased case loads on the individuals working in the system may be counter productive.

7.3.2 Installations can not support one another

Installations don't have the spare capacity, the scheduling systems, or the incentives to adequately support one another. This doesn't mean they never help one another (there are several such arrangements already), but it does mean that many opportunities to improve access could be missed.

[24]Direct care provider capacity in this simulation was set to 50% of true demand. This is probably not far from the actual state of affairs. 60-70% of utilization at most sites is in the direct care system, and PHRAMS estimates there are still sizable gaps between current staffing levels and actual need.

Scenarios 2b and 2d show that most sites simply don't have the capacity to help. Only in one instance was one installation (Site Charlie) left with enough capacity to begin serving other sites' ADSMs. Even in that case, they were able to offer only a trivial amount of help.

Many sites serve families in their direct care system, but this does not necessarily mean they have spare capacity for taking on ADSMs. Child And Family Assistance Centers (CAFACs) and School Behavioral Health programs offer a steady supply of family member care, regardless of ADSM demand. These programs and provider salaries are funded with the express intent of offering family care. It's unlikely that these carve-outs of care would be repurposed to serve distant service members.

Even if sites do have the capacity to help one another handle surges, it isn't clear they have the incentives to. MTFs are held accountable for meeting MHS's access-to-care standards (Section 2.5.1). But, as explained above, simply meeting these standards does not imply that a facility has adequate capacity. Providers or departments that don't generate enough Relative Value Units to pay for themselves are generally considered to be under performing. And, because MTFs are managed largely based on RVUs production, they have an incentive to keep all providers occupied at all times. That means that no installation is incentivized to build capacity that might one day go underutilized[25].

Most importantly, even if there were spare capacity and a site chose to share this capacity with another, they might not be able to. Today's telehealth networks rely on high-overhead coordination between sites. When a brigade returns, one installation will dedicate large blocks of provider time to another. Usually, this time is used for the wave of post deployment screenings (PDHAs) required by law, and providers then go back to serving their home installations. This offers some help. It certainly helps cope with the sharp spikes in demand for the first few days. Today, these high overhead arrangements work, but on-demand access to other sites' providers would require a scheduling system that the DoD doesn't have yet.

7.3.3 Placing variable demand on purchased care ensures insufficient capacity

It can be more difficult to grow a robust purchased care network when MTFs send unpredictable amounts of patients to the purchased care system, which is exactly what happens when MTFs serve family member demand during lulls in ADSM arrivals. MTFs and their providers need to stay busy and they have the right of first refusal for all patients. Therefore, they will rationally choose to serve more family members during periods of low ADSM demand. When they do, they send a lower than normal amount of patients to the purchased care network. Over the short term, this is great for the MTF and the MHS who keep utilization up and costs down (PC is generally more expensive

[25] It also means that a department or provider with more capacity than needed may be best served by hiding this information by delivering more care than evidence based practice would suggest. Again, this isn't a matter of intentional misapplication of resources; providers would just be doing the most good with their time. In that case, we may not even know where excess demand – and therefore opportunities to improve care elsewhere – are presenting themselves.

than DC). Over the long term, it means that purchased care providers can not rely on a steady stream of patients. Their incentives are similar to the MTFs: stay busy or go out of business.

Providers are sensitive to the rates at which TRICARE reimburses them. Therefore, we can expect them to be equally averse to a less than constant supply of patients. Only if providers are in heavily populated areas can they make up for lost TRICARE demand easily. In rural military communities, that kind of adaptation is highly unlikely. More likely is that the local cadre of civilian providers grows only large enough that all providers remain fully utilized. This leave the MTF – or the patient – to bear the consequences of a demand overload.

7.3.4 MHS and installations don't know how far behind demand they are

Without a more robust set of metrics and a better system for distributing demand, MHS and individual MTFs will not know if they are meeting the needs of their patients. By adding extra patients to a provider's caseload or by offering fewer encounters to each patient, a site can maintain access to care standards indefinitely[26]. Since these subtle forms of demand suppression are difficult to measure, it's hard to say when and where patients are underserved.

If the MHS made a substantial amount of TH available on-demand and if that care was available without excessive planning overhead, it would be quite clear exactly how many patients need care, and how many more local providers a site might make use of. Not only would MHS wind up with a better understanding of its own needs, it would better serve its patients along the way.

7.3.5 Telehealth throughput matters

If there are enough providers in a TH cell to meet their surge demand, Power Projection Platforms will require huge amounts of TH throughput. They'll need the bandwidth to allow many simultaneous connections and they'll need physical space and support staff to accomodate patients. Sites with less variance in their demand, like training centers, may not need such an overwhelming capacity.

A cursory examination of the TH usage at each site in scenario 2d shows that each MTF's TH capacity might need to be equivalent to traditional direct care capacity. Finding support staff and space at each installation would be expensive and difficult. Current TH practices (putting video conference suites in outpatient mental health facilities) might not be cost effective if it's only used to support surges every few months. The 2012 NDAA (STEP Act) would allow patients to recieve in-home TH, which could reduce the need for dedicated infrastructure at each MTF or clinic.

[26]That's not suggest anyone in the system would do this conciously. However, a responsible practitioner may triage his patients and devote more attention to the most severe cases. When they do, the less severe cases that would ordinarily recieve more attention can be neglected. Triage can be good medicine, but it indicates deficient system design. [Rosenheck and Fontana, 2007] demonstrates that increased load correlated with reduced services per patient in the VA (the study didn't establish causality).

8 Assumptions and Limitations

8.1 Demand Generation

Demand generation is backward looking. The projections made here are extrapolated from historical data. While this makes them suitable for describing trends that will repeat themselves, it may not represent the future. The methodology used to project demand here is designed to capture patterns following one type of event: deployments. It will not accurately capture care seeking that occurs at times uncorrelated with the deployment cycle, such as relocation or retirement (leaving one's combat unit and support network is stressful and many service members seek disability when separating from the services).

The United States is withdrawing from Afghanistan and has officially withdrawn from Iraq. Because fear of letting one's unit down is an aspect of stigma [Hoge et al., 2004], it is concievable that more service members will seek care once they knew they will no longer deploy. Care seeking after a service member returns from the deployment he believes to be his last may differ substantially from care seeking during wartime, and the results here may not describe those trends accurately.

The model only accounts for active duty service members and their dependents. Guardsmen, reservists, retirees and their families are not accounted for. It is difficult to tell, using our data set, when reservists are eligible for military care. It is impossible to tell when they are eligible for their own civilian health insurance. Therefore, we did not attempt to determine their populations, and could not project the demands they would place on the system.

Demand projections assume care seeking is exogenous Demand projections in this model are not affected by increased or decreased access to care in the simulation. This is intentional. Because demand isn't reduced by insufficient care in the model, we can see the amount of care that would be required to serve all patients that would want care.

The demand generation methodology is based on an Army-centric view of the deployment cycle. Most of the interviews and field work informing this thesis focused on the Army. Based on Army interviews, the demand generation algorithm explicitly focuses on return from each deployment. The algorithm only takes into account the cardinality of multiple deployments (i.e. demand after the first, second and third deployment). While this illuminates important care seeking behaviors in the Army, it may not be appropriate for every service.

Anecdotal evidence from PTSI team interviews with Marines suggest that their care seeking behaviors change substantially after what they expect will be the last deployment for a while. Marines go on shorter deployments (usually six months) and spend less time at home between deployments. Because Marines aren't home as long, they may not seek care between deployments in the same way as soldiers. A more complete analysis would also examine care seeking after a Marine's last deployment (e.g. care seeking after the last deployment, care seeking after all other deployments).

Demand generation does not completely account for care availability This demand generation scheme only records actual utilization of care. This implicitly assumes that all patients' access is limited similarly and that limits on care are similar over the deployment cycle. Under those assumptions, historical usage trends would reflect the patterns, if not the absolute magnitude, of the demand for care (e.g. if it were always the case that half of those seeking care ever got it, our pattern would be exactly correct, but our magnitude would be off by a factor of two).

In reality, fluctuations in demand will cause the system to be more over burdened at particular times (during surges). Service members and their dependents could also have better access than one another. Service members always get the first shot at MTF care, but they also may not have been referred to the network when the MTF was overburdened.

Actual demand patterns will differ from installation to installation Utilization, our proxy for demand, is affected by screening practices, outreach efforts, and care capacities. Many of these practices have changed over time and vary from installation to isntallation. Demand projections derived from service wide utilization rates over FY03-09 may not be representative of every installation. This model uses the service-wide trends because installation to installation variation within services is probably undesirable. Where installation to installation variance in demand patterns is observed, the sources of variation should be examined. If the care system is affecting these patterns, installations should consider adopting best practices from other sites or sharing practices that they have experimented with.

Demand projections capture patterns but not magnitude of care needed. The models discussed in Section 2.4 estimate the magnitude of care needed, but not the pattern in which the care is needed. The approach used here projects demand by extrapolating from past utilization for a subset of the population, and is not suitable for estimating the total demand that a site should see.

Recorded catchment areas are used as a proxy for actual location. In the demand generation process, a service member is assumed to be deployed or at their home station at all times. Using records of their TRICARE enrollment and their deployment histories, we infer their location. In some cases, a service member will have one location recorded in a given fiscal year and another recorded the next. When no relocation date is specified, we assume that they move between the fiscal years. This does not impact the utilization or arrival rate calculations, but may impact the projected population shifts at each installation.

8.2 Simulation

The simulation assumes that direct care systems are inflexible Excess demand can manifest itself in several ways. Patients can each get fewer encounters or get them less frequently than evidence based practices would suggest. Patients could get group therapy when individual therapy is more appropriate for their case. Providers could work overtime. Or, excess patients could be sent to recieve care in another system (telehealth or purchased care). In the real world, a combination of these would prevail. Though sites may engage in a combination of these behaviors, it is not clear to what extent installations do or should rely on each of these tactics.

To account for this uncertainty, the simulation takes a coarser view of the system. All of these behaviors – except for referrals to outside providers – are held constant. Holding care provision behaviors constant approximates an ideal situation where all patients recieve evidence based medicine regardless of the needs of other patients. Under this assumption, all excess load goes to the TH network or purchased care system.

This approach is desirable for this analysis even if it's not representative of the real world. By directing all excess load onto the purchased care system in the simulation, we can see the true excess demand facing each site.

The simulation doesn't prioritize patients based on acuity. The simulation does not account for any nuanced behavior within sites. All patients are seen with the same frequency and accepted with the same probability. Therefore, this model can not credibly estimate the different impacts of care shortages on patients with particular acuities.

The simulation assumes the distribution of encounters per episode does not vary across the deployment cycle. At some times at some sites, post deployment screenings are conducted by behavioral health providers. The large spikes in the first few days after Army and Air Force deployments might be attributable to these screenings. If so, the patients arriving to the system of care in the first few days would be less likely to consume a long string of encounters simply because screenings also include healthy patients who need no more care.

If screenings are unevenly distributed across the deployment cycle, then not all patient arrivals should draw their need for care from the same distribution. Those arriving in the first few days in the case above should be more likely to need only their first encounter (the screening), while those later on would be more likely to need multiple encounters.

The simulation assumes that ADSMs and dependents are indifferent to care modality. This model assigns service members to in person or TH care without regard for their preferences. It also sends most family members to the purchased care system and denies them the choice to use telehealth. Of course, real systems are more complex.

During screenings at one installation, service members are offered a choice between modalities. According to the staff, patients generally prefer the modality with the shortest waiting time. Just as some direct care capacity is carved out for families (e.g. CAFACs), some TH capacity has been built just to serve families. The School Behavioral Health program, for example, uses videoconferencing to connect young students with behavioral health providers.

Any real telehealth system will need to account for patient choice in the matter. If too many patients are averse to this modality or too many patients' psychological conditions aren't appropriate for TH, such a system may not be as useful as it appears.

All patients get weekly encounters. In the real world, patients get encounters with varying frequency. The mean time between encounters is roughly 13 days but it varies widely[27]. The simulation assumes that all patients get one encounter per week until their need has been met.

[27] Based on the same analysis presented in 4.4.

9 Future Work

Account for the variation between services. The magnitude and the pattern of care recieved by service members in each branch of the military differed substantially in fiscal years 2003 through 2009. We can not readily explain this variation. Research should elucidate the causes for this variation. Where they are caused by care system design (as opposed to say, differing amounts of combat exposure), the practices causing variation should be scrutinized and opportunities for adopting best practices should be explored.

Update this analysis to reflect mandatory PDA/PDHA/PDHRA screenings. With the responsibility to conduct new screenings and to treat the patients identified, we can expect that installations see substantially different demand patterns now than they did in our dataset (FY03-FY09). In the 2010 National Defense Authorizaiton Act [?], Congress required the DOD to provide face-to-face screenings of every service member at specific intervals after their deployment in support of contingency operations. According to the NDAA, these screenings are intended to identify service members in need of treatment for PTSD or other mental health issues. It also specifies that screenings may be conducted by mental health professionals if that wouldn't interfere with higher priority tasks. Indeed, some sites our team has visited use mental health providers for this duty either by choice or because of regional policy.

A repeat of this analysis on a fresher dataset covering the time since the new screenings were implemented would help to better understand the current demands placed on the system.

Explore TH architectures that mesh well with Embedded Behavioral Health. The Army recently announced expansion of its promising Embedded Behavioral Health (EBH) architecture [Carabajal, 2012]. Embedded Behavioral Health brings providers out of the hospital and into small clinics where a team of seven providers (plus support staff) serves each Brigade Combat Team (BCT) of several thousand Soldiers.

An Army Public Health Command report found that brigades with access to EBH to have lower inpatient admissions, exhibit fewer high risk behaviors, and to have fewer non-deployeable soldiers than brigades which recieved conventional in hospital care [Piver-Renna, 2010]. According to the Army, because providers build better relationships with unit leaders and Soldiers, they are more effective at reducing stigma and supporting the BCT's mission.

The Embedded Behavioral Health model is in tension with this thesis' recommendations. Embedded Behavioral Health relies on strong personal relationships to help build trust, while telebehavioral health (when used for load balancing) uses providers to support many installations on an as-needed basis. It is unlikely that TH providers will be able to forge such strong relationships with commanders and soldiers outside of the clinical setting. It may also be the case that EBH teams are not able to meet the changing demand from their small population. This thesis has shown that being able to pool provider capacity across several army installations could improve access to care. Meanwhile, the EBH architecture will shrink the pool size from a whole installation down to a particular unit.

The Embedded Behavioral Health architecture's capacity to handle the surges documented here should be very carefully analyzed. Where appropriate, TH should be considered as one means to support EBH teams as they deal with varying demand for care from their brigade.

Use utilization time series as a leading indicator, especially for emerging conflicts and duties. New experiences will put new stresses on service members. By looking at the time-series demand from those first exposed, better predictions of future demand can be made. For example, while the first units to return from combat in Afghanistan may not have overwhelmed the systems of care at their home installations, watching their care utilization over time could have informed us that when the majority of the military is deploying, we will see large surges in demand for mental health services. Carefully analyzing the care seeking behaviors of the first persons exposed to a new set of stressors can serve as a leading indicator.

The mental health care utilization of drone pilots has not yet been well characterized. It is possible that they too seek care in distinct patterns (for example, after using deadly force). Many operators have been serving for years, and because they do not deploy and return in large groups, new pilots and crew members can be added and removed from that duty in a regular pattern. This regular pattern of arrivals and departures all but guarantees a steady demand for services (no surges), but might obscure important leading indicators about the mental health risks of remote combat.

Data on the mental health utilization of current and past drone pilots should be carefully analyzed to assess the impact of their duties on their utilization of mental health services. While this is of little value for clinical operations, it will be valuable to creating a system of care that effectively supports a new breed of service member through their military lifecycle.

Use this knowledge for more than operations. This analysis demonstrates the use of historical demand patterns to improve hospital and clinic operations. The knowledge gleaned from this analysis can be used outside of the clinical care delivery system. For example, unit leaders in the Army should be aware that their Soldiers may struggle with mental health issues at particular points in the deployment cycle. Leaders can use this knowledge to make better decisions when planning training missions, leadership turnover, and permanant changes of station. Organizations serving military communities (Chaplains, Army Family Services, Fleet & Family, etc.) can use this information to anticipate emerging non-medical needs which may accompany increases in mental health utilization.

Enhance the predictive quality by looking at more variables. This analysis takes a more fine grained view of the deployment cycle than other models, but it is by no means an exhaustive analysis.

Service members with different Military Occupational Specialties (MOSs), duties, and experiences may have different demands for care. Some service members see constant combat for the duration of their deployment while others may stay in forward operating bases where they see little or no direct combat. Some MOSs are known to be especially stressful, like detainee operations [J-MHAT, 2011].

The demand generated by a particular unit is likely correlated to that unit's casualties and combat exposure. Using information from databases like AHLTA-Theater (the military's in theater medical records system) and the Combined Information Data Network Exchange (CIDNE), a demand prediction based on combat exposure could help sites plan for larger than normal surges months before a unit returns. Unit type and mission might also impact the rates at which service members seek care.

Improving demand predictions should focus on phenomena that will affect the nation wide demand for care. A robust telehealth network like the one described in Section 10 could easily help shift demand from one site to another, but if the nation wide need for care varies drastically, moving supply and demand from site to site will do little to help.

Calculate Unconstrained Demand Patterns and Volumes The methodology here is based on historical utilization. In some times at some sites, this utilization was constrained by the system's capacity. Utilization is therefore less than or equal to demand and actual reciept of care happens some time after patients request it. As a result, demand patterns here probably under estimate the real variance in patient requests for care.

Demand projections could be improved by examining utilization in cases where demand was not constrained. For example, instead of looking at all returns from theater, an analysis could examine only the returns of small units to sites which, at the time of their return, saw relatively little demand because many other units were deployed.

The demand projections here were not backtested because individual sites vary and not all information about them was available (for example, number of full time equivalent providers). If many examples of unconstrained care-seeking were identified, some could be used to calculate the expected rate of care seeking while others could be withheld to backtest such a projection scheme.

10 Policy Recommendations

10.1 Develop an Efficient System for Scheduling Appointments with Telehealth Providers

Today, telehealth scheduling relies on large amounts of administrative overhead. Each site runs its own copy of the MHS's scheduling software, but providers at distant sites can not readily access it, and it is difficult to make appointments. Because the system is inflexible and inefficient, patients and providers don't get the most out of telehealth. For example, some TH cells offer a constant volume of telehealth (some number of hours per week) to each site they service. The distant site then fills the schedule themselves. In other cases, sites will support one another during a surge by arranging for a large block of support (several providers for several days) then abruptly stop when the peak of the surge has passed. Making arrangements like these require phone calls and special configuration of computer systems. Providers or their support staff go through a cumbersome process to update the patients medical records after the fact.

An efficient scheduling system would allow MTFs to make TH appointments for their patients on an as-needed basis. When they need more care, like during a surge, they can pass patients along to TH providers, and when they no longer need the care, they can free those resources for other sites to use.

With the passage of the 2012 NDAA, one provider can see patients in more locations than before. In the past, providers were required to seek credentials and clinical priveliges from every site they served, a very time consuming process. Because of this, providers served only one or a few distant sites, which helped to limit scheduling overhead. Patients and providers wont be able to take advantage of these new changes in the law until they can actually coordinate with one another to set up appointments.

10.2 TH Cells Should Provide Surge Support

A cadre of providers dedicated to supporting overburdened sites will be more effective than providers whose first priority is serving their home installation. Providers at most sites have more patients than they can serve, even during lulls in demand. Simply connecting MTFs to one another will not be enough. Providers are busy, and might be expected to use any lull in demand to increase the quanity of care provided to their own service members or their dependents before reaching out to help patients at distant sites. A group of providers dedicated to helping the most overburdened sites will not face this conflicting choice. They can distribute their time to the sites most in need at any given time.

Reliable surge support will leave sites with a more constant ratio of supply and demand, which can be managed more effectively. If a site is consistently over burdened, it can hire a new provider. If the same site faced variable demand without surge support, it couldn't hire a new provider, since his utilization might fall below 100% during lulls in demand. Installations can also depend on their local purchased care systems to absorb a constant supply of patients, but not a highly variable one. Local practices wont stay solvent if they can only fill their schedule six months out of the year, so local networks will tend to support the minimum number of referrals generated by a local installation.

Dedicated TH providers (a "TH cell") could be located in tertiary care facilities, in more attractive labor markets, or distributed across many traditional installations. Today, most TH cells are located at a regional headquarters within large tertiary care facilities. Because they are housed within the tertiary care facility, they are managerially independent from the needs of any one site and are not monopolized by their home MTF.

Distributing providers across the installations they would support instead of grouping them in a central location would allow them to offer in-person services when need for telehealth drops off or when their own site is in a surge. With the scheduling system described in Section 10.1, it would be possible for designated providers at conventional installations to open their appointment schedules to all military patients around the world with equal priority. Of course, such providers would have to be specially designated, as local leadership is unlikely to surrender resources without direction from command. Service level medical commands could direct a certain proportion of providers at each site be designated as TH providers. Less controversially, the services could fund additional provider positions at each site which would be automatically designated as TH providers.

Dedicated TH providers could also be hired in attractive locales. Many rural sites have trouble attracting providers. By establishing cells in major metropolitan areas, the military could tap into a larger labor pool.

10.3 Responsibility to Deliver Access Should Not Belong to the MTF

Service level medical commands should assume responsibility for delivering access to care. Incentives will need to be realigned and responsibilities shifted. In the past, pushing responsibility for outpatient behavioral health care down to MTF's and the regional TRICARE contractor made sense. With the opportunity to organize a flexible system for supporting overburdened sites, each service should take responsibility for meeting MHS access to care goals and delegate a new set of responsibilities to sites.

MTFs are saddled with variable demand and an effectively constant capacity for care provision. Right now, responsibility for meeting access-to-care standards and for meeting RVU expectations belongs to the MTF. To meet those access expectations during surges in demand, someone somewhere gets the short end of the stick. Providers could be overworked, patients made to wait or given less care, or families referred to the network could have difficulty getting appointments if they get one at all.

Sometimes, hospitals have to make due and deliver the best medicine they can under the circumstances. The increasing feasibility of telehealth changes those circumstances. No evidence reviewed in the course of this work suggests that MTFs or providers aren't doing their best to deliver access to quality care. But, best practices from yesterday won't make sense in a world where MTFs can frictionlessly offload excess demand to distant providers (or pitch in and help distant patients when things are slow).

MHS's Guide to Access Success [MHS, 2008] shows that predictable and consistent demand allow the system to better deliver access to care without sacrificing in other ways[28]. Section 7 shows that aggregating the demand for care across many Army installations results in a more manageable demand pattern – even with a constant supply of providers.

Aggregation at the Service Level is Most Promising Aggregating demand at the service level is most important for the Army, but would likely benefit all of the services. The Army has seen surges in demand for many years, while some services, like the Marine Corps, may not have. But, the mandated screenings in the 2010 NDAA should increase the variability of demand for all services, especially immediately after return. Even for services which do not see surges in demand today, building a robust system is important. In addition, future wars may place different burdens on each service.

Aggregating demand at the service level offers more advantages and fewer challenges than at a regional or joint level. Regional level aggregation could prove useful, but offers no major advantage to service wide pooling. Regional pooling could result in unwarranted variation in practices and technologies making it difficult to combine efforts in the future. Demand could be aggregated at a joint level, but this would present problems with culture and with accountability. Service members often prefer uniformed providers who understand military culture [Arthur et al., 2007]. This preference may also extend to the service level. At least one officer interviewed mentioned that service members in his charge resented seeing providers from another service who they did not feel understood the challenges they face.

Responsibility Requires Authority and Means Responsibility for access to care should reside in the same place as the authority and means to deliver it. Today, each of the services hold the budget authority for their continental MTFs and are accountable for their own beneficiaries[29]. If a service level medical command takes over the responsibility for access-to-care, they must also be able to hire the requisite staff in TH provider cells or at individual installations.

[28] This is explained in the context of Open Access appointing. MHS explains Open Access as "...a patient will see a provider on the same day that they request an appointment." The guide cites many advantages including reduction in time consuming triage, enhanced continuity of care, and increased patient trust of the sytem, but MHS does not endorse it. First on the list of disadvantages is that "Surges in demand may require that providers and support staff work extended hours". Second, that "Staff burn out is possible".

[29] There are exceptions, like the Joint Task Force National Capital Region Medical, which supports all services in the Washington, DC area.

In the near future, the new Defense Health Agency (DHA) will take control of MTFs in multi-service markets[30] [MHS, 2011], but single service market MTFs will remain under the budget authority and responsibility of their service. Under the new architecture, multi-service markets will get their own budget and be responsible for their own capacity planning. Administrators of these markets should make an explicit agreement with their MTFs' parent services to ensure that the TH system is not used as a source of "free" care for the market, which would place an undue burden on the service's budget. Through manpower or budget, multi service markets should contribute as much to the TH network as they reap.

10.4 Close the Loop Between Changes in Demand and Changes in Capacity

New providers should be automatically and quickly hired when demand outpaces supply. Today, measuring the difference between supply and demand is nearly impossible. When new providers are brought on board, hiring, credentialling and priveliging them takes too long. This recommendation does not strictly require the adoption of the preceeding ones, but they do complement one another.

Credibly Measuring Excess Demand Today, each installation reports the number of appointments meeting or failing to meet MHS's access to care standards. Departments within MTFs also report their productivity (in Relative Value Units). This productivity is then compared to the budget each department recieves. As a result, departments and providers are incentivized to get patients in to the system in a timely manner and to always stay busy. Both are important goals. But, since not all aspects of access and quality are as rigorously measured, patients and providers may be left carrying an undue burden that is not prominently reflected in installation level metrics. Patients could recieve less care, group encounters when evidence suggests they would be best served by invidual ones, less frequent encounters, or providers could end up overworked. None of these is desirable, and all eventually exact a price on patients and on the military.

MHS must be able to confidently measure the supply and demand facing each pool of providers (either at a hospital/clinic level or a network of installations sharing resources via telehealth). The analysis in this thesis suggest that it unlikely that an Army installation can maintain 100% provider utilization and offer access to a full complement of care to all newly arriving patients.

The difference between supply and demand can be accurately measured in at least ways:

1. Improve the measurement of all aspects of care provision to ensure all patients recieve evidence based treatment

2. Project "true demand" and compare it to current supply (e.g. PHRAMS, ASAM)

3. Offer installations the ability to send patients for whom they can not deliver a full course of evidence based treatement to a telehealth provider

[30] Multi Service Markets (MSMs) are small areas in which more than one service has an MTF

The first of these options requires surveillance of clinical decisions. Sites would need to show that they are not delivering too little care or too much care per patient. It would be very difficult to implement properly and would require substantial manpower to assess provider decision making. Such manpower is already in short supply.

The second option requires a precise calculation of true demand. While patterns can be predicted, absolute magnitude is difficult to discern. Neither volume of requests for care nor workload models offer a robust estimate of demand for mental health care. Demand for care can be measured by appointment requests in some specialties. But, in mental health, patients recieve a string of encounters after entering the system. The length of treatment depends heavily on provider judgment. Appointment requests can only tell us how many patients are looking to begin a new episode of care, but they can not tell us the true demand for encounters. Models which estimate true demand (e.g. PHRAMS) rely on assumptions about the amount of care each patient should get and of the amount of suppressed demand. Care per patient varies by case and suppressed demand is notoriously difficult to measure.

The last option is preferable because it frees the department chiefs and providers from balancing access and quality when both are not fully acheivable. Patients can get timely care from TH providers and no patient will be underserved once they do enter the system – so long as there is sufficient capacity somewhere in the TH network. Ensuring sufficient capacity, especially at the outset will be difficult.

Turning Metrics into Manpower Several interviewees roundly criticized the hiring process for its length and difficulty. It can take over a year to go from advertising a position to having a new providers treat patients. When demand does outpace supply, hiring actions must be quick and automatic. They should not need to wait for new budgeting cycles or for individual approval. Each service should authorize the conditional hiring of a substantial number of telehealth or local providers according to demand measurements.

Services should take care to look into the future and consider using the same metrics to systematically shrink the mental health provider work force (or transition it to the Veterans Affairs system) when and if demand falls off in the years following the wars in Iraq and Afghanistan.

11 Conclusion

Military hospitals see drastic variations in demand for mental health services. The departures of large combat units lowers demand and the units' subsequent retuns increase demand by a much greater margin than their headcount would imply. Service members, especially those in the Army and Marine Corps, seek more care after deployments and they enter care in pronounced patterns. Keeping up with these patterns is difficult for MTFs which ordinarily rely on the relatively inflexible supply of care generated by their local staff.

Telehealth can improve access to care at installations that face these variable demand patterns. This thesis shows how appropriate incentives and technological systems can help the services balance their demand across a nation wide group of providers. With enough providers dedicated to surge support and the means to efficiently schedule telehealth appointments, installations facing a surge in demand do not have to put undue burden on their patients, their providers, or the local purchased care network.

Telehealth is not a panacea. In a clean mathematical analysis, it can help to optimize care delivery across the system, but it does not offer all of the benefits of other architectures. In person providers can build rapport with service members before they seek care and engage in public health projects that would be impossible to accomplish by telehealth. Telehealth must complement existing efforts to reduce stigma and improve relationships with commanders.

New laws and new technologies make telehealth especially attractive. The military – especially the Army – should make sure that the ongoing proliferation telehealth technology is complemented by appropriate policy decisions.

Appendices

A Description of M2 Data Set Used in the Analysis

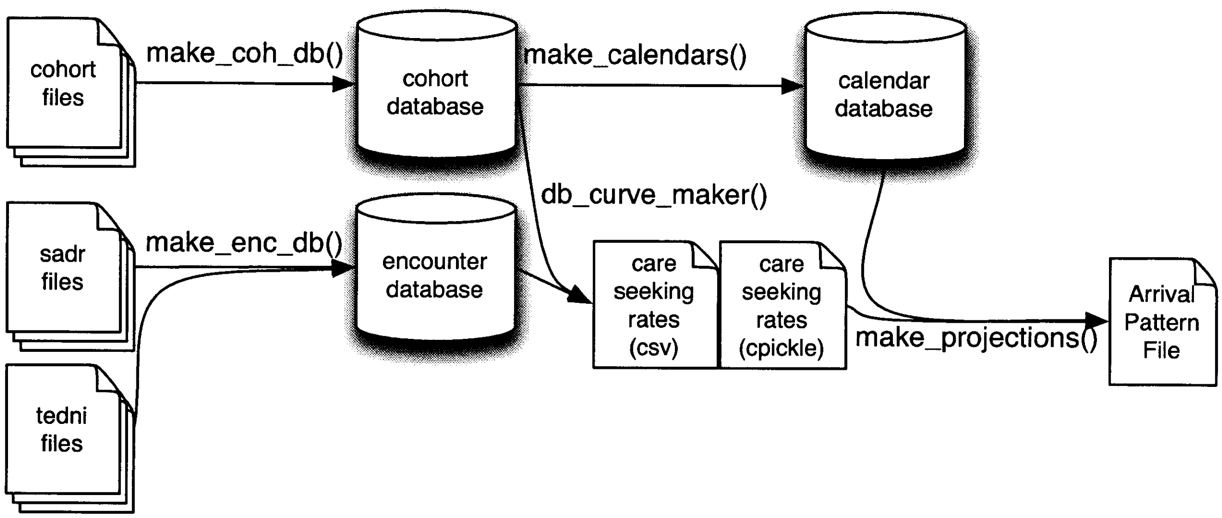
MHS provided a version of the M2 as a set of .sas7bdat files (a specially g-zipped SAS file which can usually only be decompressed by SAS). So that they could be used in any analysis softward they were converted these to .csv files. The first row of the csv file were the headers (matching the field names in the MHS's M2 data dictionary, which can be found easily on the internet. The M2 data dictionary contains more complete descriptions than those presented below, and should be used whenever trying to analyze the data. Each file then contained millions of lines of data, each representing a person (cohort files) or an encounter (tedni, tedi, sadr, and sidr). MHS also provided pdts files, containing prescription information, but those were not used in this analysis.

B Source Code for Correlating Utilization Rates and Arrival Rates to the Deployment Cycle

B.1 Summary

Figure 50 shows a simple flow of data analysis going from the *.csv files to the care utilization and arrival rate files (see Section 4) and the Arrival Pattern File used in the ARENA simulation (see Section 5).

Figure 50: Summary of Data Analysis



B.1.1 A note on practicality

The original set of .sas7bdat or the created set of .csv files combined to be tens of gigabytes in size. In order to work with them efficiently, the algorithms described here bring the datasets into memory as databases with appropriate indices.

Because it can take so long to process some interim products, and because they were used to do research as these algorithms were being written, interim data products are often stored. This sped up debugging and quicker turnaround on slightly changed versions of the analysis.

B.1.2 Creating Service-Specific Subdirectories

A special subdirectory was made for each of the four services discussed in this thesis using the cohort_maker script (see Section B.2.5). This subfolder contained a copy of each of the .csv files used in this analysis that included only lines which referred to individuals who, at some point, were associated with the service in question (e.g. a "FullArmyData" folder which contained all lines from the .csv files for anyone in any family associated with the Army).

While we could have filtered records as needed in other algorithms, this also created databases of a manageable size for our workstation.

B.1.3 Intermediate Databases

A script, *dbmaker.py* (source code in Section B.2.2) runs two functions, make_coh_db() and make_enc_db(), which each create the respective databases (a third function creates a more detailed encounter database, this analysis does not use that database). The cohort database contains all lines from all the individual cohort files (there are one for each FY 2003-2009). All rows can be uniquely identified by the person's scrambled ID (scram_edipn) and the year of the cohort file (one column in the database). Because not all data recorded in the cohort.csv files was required for this analysis, the database table takes only takes in the following fields:

- scram_edipn: A unique personal identifier
- famid: A family identifier
- startdep: The start of the most recent deployment, recorded as a string
- enddep: The end of the most recent deployment, recorded as a string
- BENCAT: The person's beneficiary type (ACT = Active Duty, DA = Dependent of Active Duty, GRD = Guard or Reserve, etc.)
- PARC: Person association reason code, describes relation to the family's sponsor (self, spouse, child, etc.)
- FY: The year of the cohort for this line
- service: Branch of service (A = Army, F = Air Force, etc.)
- BENCAT_CHG: The type of beneficiary change (e.g. retirement)
- AREA: The DMIS code associated with this person. Their home station in this analysis.
- AREA_DT: The date on which the person moved from one AREA to another.
- BEGELG_DT: The date on which this person became eligible.
- ENDELG_DT: The date on which this person is no longer eligible (this changes from year to year, presumably as people reenlist)
- BENCAT_DT: The date on which the person's BENCAT value changed, expressed as days since 1 JAN 1960

Encounters from both the sadr and tedni files were similarly combined into a single database table. Only encounters for which the mh_prov field was non-zero were accepted. Again, only a subset of fields were brought into the database:

- scram_edipn: a unique personal identifier
- famid: a family identifier
- begdate: the date the encounter began, recorded as days since 1 JAN 1960
- wgtd_work: the RVUs generated by the encounter
- source: either sadr or tedni, to differentiate care recieved in direct and purchased care

B.1.4 Rate Calculation

Using the encounter records, and cross referencing the ID for each with the person, and the data of each encounter with that person's values of B, P, and S for that day, the status of each encounter can be determined. By looking at the person's entire history (all of their cohort entries) we can determine, with reasonable accuracy, their values of B, P, and S on every date during their time on the military's rolls.

Dividing encounters in each combination of values of B, P, and S by the number of days beneficiaries spent in each of those combinations gives us encounter and arrival rates.

To do this, the script *db_curve_maker.py* is run. It calculates the care seeking rates for each type of beneficiary in each service. Information about the rates of each beneficiary category in each service are dumped into descriptively named *.cpickle[31] files which are used later to project arrivals for various installations.

B.1.5 Population Calendar and Demand Projections

The script *cal_sc.py* (see Section B.2.3) creates the calendar database and the projections (population, encounters and episodes) for each site specified.

The make_calendars() function from *arena_seed.py* (see Section B.2.4) is used to create a population calendar database using one or many (in this analysis, one for each service) cohort databases as inputs. This database contains the population at each of the specified sites on each day in each combination of B, P, and S.

This population calendar database is so specific about base demographics each day because it is used to project rates based on the equation in Section 4.5.

[31] cpickle is a serialized format for python data, in this case, a nested dictionary (hash table)

The script then calls the make_projections() function twice. First, it is called with the parameter "pop", which creates a .csv file which contains the ACT and DA population of each installation on each day by effectively summing all subpopulations for each installation, day, B combination. Next, the script is called to project new episodes of care, using the "epis" parameter. This time, make_projections multiplies each daily subpopulation by the rate at which it arrives to the care system (stored in the rate *.cpickle files). It creates two .csv files on this run. The first file contains the ACT and DA arrival rates for each installation on each day (these and the population .csv are used to generate the figures in Section 4.5). The second file is specially formatted to be used by ARENA during simulations.

B.1.6 Dependencies

In addition to the standard python libraries, these scripts use apsw, "Another Python Sqlite Wrapper" which simplifies loading the databases into memory.

These scripts rely on a host of other files containing helper functions and classes. For example, *new_history_hanlder.py* contains a class that makes sense of multiple cohort lines and can use them to determine a person's installation and status on any given date. All custom code created for this analysis is presented in Section B.2.

B.2 Python Files

B.2.1 Code Quality

The code here is presented as-is. Some file names are not as descriptive as they could be. Many of the design decsions will seem insane. These are the product of several months of trial and error and cobbling together of peices of code which already worked. Ordinarily, this would not constitute publishable or sharable code. But, in the interest of transparency and in in the interest of supporing others who wish to do similar work, my assumptions are presented here in gory detail.

It is my hope that this is useful for understanding exactly how the calculations in this analysis were made. In particular, this code serves to document the exact assumptions used to determine a person's location and status on a date and to exclude persons or families from analysis all together (those with multiple service member sponsors, for example). To that end, I have tried to comment these files well as they were written. There are certainly better ways to conduct this analysis, and if doing so from the ground up, it would be worth implementing a (better) architecture.

Proceed at your own risk.

B.2.2 dbmaker.py

```
# This script builds sqlite databases containing a subset of encounters
# and all cohort rows.

import sqlite3 as lite
import apsw

import sys, os, csv
from collections import namedtuple
from jth_tools import apsw_backup, Printer

def make_enc_db(location,filename,filter):
    #first, build the database in memory so it's quick to make indices
    print "making apsw memory encounter db"
    con = apsw.Connection(':memory:')

    with con:
        cur=con.cursor()
        cur.execute("DROP TABLE IF EXISTS encounters")
        cur.execute("CREATE TABLE encounters(id TEXT,\
                     famid TEXT,\
                     begdate TEXT,\
                     wgtd_work float(8),\
                     source TEXT)")
        encounters_read=0
        for filetype in ["sadr","tedni"]:
            for year in range(2003,2010):
                csv_in=csv.reader(open(location+filetype+"_fy"+str(year)[2:4]+"_scram.csv"))
                m2row=namedtuple('m2row',csv_in.next())
                for row in csv_in:
```

```
                    nt_row=m2row._make(row)
                    if getattr(nt_row,filter)!="0":
                        if encounters_read %10000==0: print "adding entry", encounters_read
                        cur.execute("INSERT INTO encounters VALUES(?,?,?,?,?)", [nt_row.
                            scram_edipn,nt_row.famid,nt_row.begdate,nt_row.wgtd_work,nt_row.
                            source])
                        encounters_read+=1
        print "Creating index by ID"
        cur.execute("CREATE INDEX byid ON encounters(id)")
        print "Creating index by familyID"
        cur.execute("CREATE INDEX byfamid ON encounters(famid)")

    #then backup that database to the hard-drive in specified location
    print "Storing database..."
    dest_db=apsw.Connection(location+filename)
    with dest_db.backup("main",con,"main") as backup:
        while not backup.done:
            backup.step(100)

def make_full_enc_db(location,filename):
    #first, build the database in memory so it's quick to make indices
    print "making apsw memory encounter db"
    con = apsw.Connection(':memory:')

    with con:
        cur=con.cursor()
        cur.execute("DROP TABLE IF EXISTS encounters")
        cur.execute("CREATE TABLE encounters(id TEXT,\
                    famid TEXT,\
                    begdate TEXT,\
                    wgtd_work float(8),\
                    dx1 TEXT,\
                    dx2 TEXT,\
                    dx3 TEXT,\
                    px1 TEXT,\
                    mh_pdx INT,\
                    mh_anydx INT,\
                    mh_prov INT,\
                    source TEXT)")
        encounters_read=0
        for filetype in ["sadr","tedni"]:
            for year in range(2003,2010):
                csv_in=csv.reader(open(location+filetype+"_fy"+str(year)[2:4]+"_scram.csv"))
                m2row=namedtuple('m2row',csv_in.next())
                for row in csv_in:
                    nt_row=m2row._make(row)
                    if encounters_read %10000==0: Printer("adding entry "+str(
                        encounters_read))
                    cur.execute("INSERT INTO encounters VALUES(?,?,?,?,?,?,?,?,?,?,?,?)", \
                                [nt_row.scram_edipn,\
                                 nt_row.famid,\
                                 nt_row.begdate,\
                                 nt_row.wgtd_work,\
                                 nt_row.dx1,\
                                 nt_row.dx2,\
                                 nt_row.dx3,\
```

```
                                  nt_row.px1,\
                                  nt_row.mh_pdx,\
                                  nt_row.mh_anydx,\
                                  nt_row.mh_prov,\
                                  nt_row.source])
                    encounters_read+=1
        print ""
        print "Creating index by ID"
        cur.execute("CREATE INDEX ebyid ON encounters(id)")
        print "Creating index by familyID"
        cur.execute("CREATE INDEX ebyfamid ON encounters(famid)")

        #now add all cohort information
        cur.execute("DROP TABLE IF EXISTS cohorts")
        cur.execute("CREATE TABLE cohorts(\
                    id TEXT,\
                    famid TEXT,\
                    startdep TEXT,\
                    enddep TEXT,\
                    BENCAT TEXT,\
                    PARC TEXT,\
                    FY INT,\
                    service TEXT,\
                    BENCAT_CHG INT,\
                    AREA INT,\
                    AREA_DT INT,\
                    BEGELG_DT INT,\
                    ENDELG_DT INT,\
                    BENCAT_DT INT)")
        individuals_read=0
        for year in range(2003,2010):
            csv_in=csv.reader(open(location+"cohort_fy"+str(year)+"_scram.csv"))
            coh_row=namedtuple('cohort_row',csv_in.next())
            for row in csv_in:
                nt_row=coh_row._make(row)
                if individuals_read %10000==0: Printer("adding entry "+str( individuals_read
                    ))
                cur.execute("INSERT INTO cohorts VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?)", \
                            [nt_row.scram_edipn,\
                             nt_row.famid,\
                             nt_row.startdep,\
                             nt_row.enddep,\
                             nt_row.BENCAT,\
                             nt_row.PARC,\
                             nt_row.FY,\
                             nt_row.service,\
                             nt_row.BENCAT_CHG,\
                             nt_row.AREA,\
                             nt_row.AREA_DT,\
                             nt_row.BEGELG_DT,\
                             nt_row.ENDELG_DT,\
                             nt_row.BENCAT_DT])
                individuals_read+=1
        print ""
        print "Creating index by ID"
        cur.execute("CREATE INDEX cbyid ON cohorts(id)")
```

```
        print "Creating index by familyID"
        cur.execute("CREATE INDEX cbyfamid ON cohorts(famid)")
    #then backup that database to the hard-drive in specified location
    print "Storing database...",location,filename
    dest_db=apsw.Connection(location+filename)
    apsw_backup(con,dest_db)

def make_coh_db(location,filename):
    #first, build the database in memory so it's quick to make indices
    print "making apsw memory cohort db"
    con = apsw.Connection(':memory:')

    with con:
        cur=con.cursor()
        cur.execute("DROP TABLE IF EXISTS cohorts")
        cur.execute("CREATE TABLE cohorts(\
                    id TEXT,\
                    famid TEXT,\
                    startdep TEXT,\
                    enddep TEXT,\
                    BENCAT TEXT,\
                    PARC TEXT,\
                    FY INT,\
                    service TEXT,\
                    BENCAT_CHG INT,\
                    AREA INT,\
                    AREA_DT INT,\
                    BEGELG_DT INT,\
                    ENDELG_DT INT,\
                    BENCAT_DT INT)")
        individuals_read=0
        for year in range(2003,2010):
            csv_in=csv.reader(open(location+"cohort_fy"+str(year)+"_scram.csv"))
            coh_row=namedtuple('cohort_row',csv_in.next())
            for row in csv_in:
                nt_row=coh_row._make(row)
                if individuals_read %10000==0: print "adding entry", individuals_read
                cur.execute("INSERT INTO cohorts VALUES(?,?,?,?,?,?,?,?,?,?,?,?,?,?)", \
                            [nt_row.scram_edipn,\
                             nt_row.famid,\
                             nt_row.startdep,\
                             nt_row.enddep,\
                             nt_row.BENCAT,\
                             nt_row.PARC,\
                             nt_row.FY,\
                             nt_row.service,\
                             nt_row.BENCAT_CHG,\
                             nt_row.AREA,\
                             nt_row.AREA_DT,\
                             nt_row.BEGELG_DT,\
                             nt_row.ENDELG_DT,\
                             nt_row.BENCAT_DT])
                individuals_read+=1
        print "Creating index by ID"
        cur.execute("CREATE INDEX byid ON cohorts(id)")
        print "Creating index by familyID"
```

```
        cur.execute("CREATE INDEX byfamid ON cohorts(famid)")

    #then backup that database to the hard-drive in specified location
    print "Storing database..."
    dest_db=apsw.Connection(location+filename)
    with dest_db.backup("main",con,"main") as backup:
        while not backup.done:
            backup.step(100)



if __name__ == '__main__':
    if len(sys.argv)>1: sample_dir=sys.argv[1]+"/"
    else: sample_dir=""
    location="/Volumes/Data/DataCube/nas/data/batch_csv/"+sample_dir
    locationPC="Z:/DataCube/nas/data/batch_csv/"+sample_dir
    if os.name=="nt":
        location=locationPC

    #make separate databases so we can pull them into memory individually if
    #we so choose

    fdbname=sample_dir[:-1]+"-outpatients.db"
    make_full_enc_db(location,fdbname)

    filter="mh_prov"
    edbname=filter+"_encounters.db"
    if edbname not in os.listdir(location): make_enc_db(location,edbname,"mh_prov")
    else: print "found the encounter database"

    cdbname="cohorts.db"
    if cdbname not in os.listdir(location): make_coh_db(location,cdbname)
    else: print "found the cohort database"
```

B.2.3 cal_sc.py

```
"""
This script makes calendars for the relevant installations, then loads the rates
calculated previously, and multiplies these by one another to produce two forms
of projections.  First, it creates a csv file that can be used to visualize
arrivals.  Second, it creates an csv file that can be used to seed the ARENA
simulation
"""
import os, sys
from optparse import OptionParser
from arena_seed import make_calendars, gather_rates,make_projections

if __name__ == '__main__':
    parser=OptionParser()
    parser.add_option("-v","--verbose",action="store_true",dest="verbose",default=False)
    parser.add_option("-t","--temp",action="store_true",dest="use_temp_dir",default=False)
    parser.add_option("-s","--subfolder",action="store",type='string',dest="sample_dir",
        default="")
    (options,args)=parser.parse_args()

    sample_dir=options.sample_dir
    verbose=options.verbose
    use_temp_dir=options.use_temp_dir

    location=""
    if use_temp_dir:
        location = ('/Users/johnhess/Desktop/DataCubeTemp/' + sample_dir+"/")
    else:
        location = ('/Volumes/Data/DataCube/nas/data/batch_csv/' + sample_dir+"/")

    if verbose: print "location is",location

    services=[]

    #full service cohorts
    services=services+["FullArmyData/",\
                "FullNavyData/",\
                "FullUSMCData/",\
                "FullUSAFData/",]

    FY09=(14153,14516)
    FY0809=(13788,14516)
    FY0709=(13423,14516)

    #make calendar for every cohort entry ever
    source_cohorts=[location+service+"cohorts.db" for service in services]
    #source_cohorts=[location+"cohorts.db"]

    installations=[] #put in the DMIS numbers that you need to project here

    #actual site DMIS id's redacted

    #get the handle of a database containing all the stuff we need to konw
    cal_db=make_calendars(FY0709,source_cohorts,installations,to_mem=True,
        new_calendar_database=location+"calendar.db")
```

```
    #create rate dictionaries from many files in a single directory
    #and put them in a database of their own
    rate_location=location+"RatePickles/"
    rate_db=gather_rates(rate_location)

    #make projections as an arena seed
    #make csv files for installations of interest showing population and
    #diagnosis rates
    make_projections(cal_db,rate_db,installations,"pop",location)
    make_projections(cal_db,rate_db,installations,"epis",location,deterministic=False)
```

B.2.4 arena_seed.py

```
"""
This file contains functions which can be used to project demand at various
installations.  Demand, in this case, is the number of arrivals to the system
(new patients).

To do this, it multiplies the care demand rate for each type of
person by that population of similar people.  There is a unique rate for every
combination of personal characteristics:

    S (service): A=Army, N=Navy&Marines, F=Air Force
    B (bencat):  ACT, DA, GRD, DGR...
    P (position in dep cycle): 'pre'=has never deployed, 'during'= is deployed
        integers 1-365: days since deployment, 'long since'=more than 365 days
        since returning from deployment

For service members, the rate corresponds to their position in the deployment
cycle.  For dependents, the rate corresponds to the position of their sponsor
in the deployment cycle.

So, for a family of 2, an Army Active Duty husband and his wife, on the 10th day
after returning from a deployment, their characteristics are:
    Husband:
        S='A'
        B='ACT'
        P=10
    Wife:
        S='A'
        B='DA' (dependent of active duty)
        P=10

For an installation, the total new arrivals on each day for each bencat is the
sum of rate(S,P)*population(day,S,P) for that bencat.

This file contains 2 main functions:

calendar_maker(...):
    read all three of the cohort databases
    and produce demographics/populations for each represented installation for
    each day in the specified window.

arena_seed(...):
    uses these calendars and known rate informaiton to create a seed for the
    arena model.  Rate information is derived from the "...rates.cpickle" file
    for the relevant bencat/service
"""

#python libraries
from collections import namedtuple
import cPickle
import csv
import os.path
import datetime

#other libraries
```

```
import apsw #SQLite, but with ability to backup db into memory
from numpy.random import poisson

#personal tools
from jth_tools import Printer, apsw_backup, make_translator
from db_tools import *
from new_history_handler import history

#globals
verbose=False

def make_calendars(window, cohort_db_filenames,installations, to_mem=True,
        new_calendar_database=False):
    """
    Returns the connection handle to a database with the population of each
    installation by day for the window specified.

    The dictionary returned has five-tuple keys:
        (Installation DMIS ID, date, service, bencat, Position in deployment cycle)

    The value of each entry is the number of people in that combination assigned
    to that installation on that day.

    Accepts:
        window
            a two-tuple window (two integers representing days since Jan 1 1970)
        cohort_db_filenames
            cohort_db_filenames a list of filenames representing all of the cohort
            databases that should be represented in the calendar.
        installation
            a list of the dmis ids that we want to make cals for
        to_mem
            whether or not to load the dbs to memory for faster computation
        new_calendar_database
            either a filename or False. If a filename, creates a SQLite database in
            that location with the dictionary entries each occupying a row.
            Each member of the key tuple is a column, and the value is another


    Generally, there will be one filename for each of the
    Full*Service*Data files.  Because each of those folders contains all
    families who ever have a member associated with that service, a family
    with one Soldier and one Marine would end up in both folders.  To make sure
    we don't double count such families, we keep a running set of families
    accounted for.
    """
    print "making calendars"
    calendar={} #an empty dict
    recorded_families=set([]) # an empty set

    #see if the file already exists.  if it does, load it into memory and return
    #its connection handle
    if os.path.isfile(new_calendar_database):
        print "Found the database, loading it to memory"
        disk_db=apsw.Connection(new_calendar_database)
        mem_db=apsw.Connection(":memory:")
```

```
        apsw_backup(disk_db,mem_db)
        return mem_db

    else: print "Creating new calendars from scratch... this can take a while"

    #create a database to track the populations at each installation
    _cal_db=apsw.Connection(":memory:")
    cal_curs=_cal_db.cursor()
    cal_curs.execute("DROP TABLE IF EXISTS calendars")
    cal_curs.execute("CREATE TABLE calendars(\
                                            AREA INT,\
                                            day INT,\
                                            service TEXT,\
                                            BENCAT TEXT,\
                                            status TEXT,\
                                            persons INT, PRIMARY KEY (AREA, day, service, BENCAT, status))")
    #don't include status, we want to be able to pull all rates in the later routine

    #for each of the files, add its information to calendar and families
    for filename in cohort_db_filenames:
        print "Aggregating information from",filename
        #establish a database connection
        _coh=apsw.Connection(filename)

        #if to_mem, load a version of the databse in memory to make things quicker
        coh=None
        if to_mem:
            coh=apsw.Connection(":memory:")
            with coh.backup("main",_coh,"main") as b:
                while not b.done:
                    b.step(10000)
                    Printer("Loading: "+str(100.0*b.remaining/b.pagecount)+" percent remaining")
                print ""
        else: coh=_coh

        curs=coh.cursor()

        #create a tuple for use in indexing cohort database rows
        #get the cohort columns
        coh_columns=get_columns(curs, "cohorts")
        #define a named tuple for the format of the cohort database
        cohort_row=namedtuple("cohortdbrow",coh_columns)

        #determine all families represented
        families_in_db=[]
        installations.sort()
        for inst in installations:
            families_in_db+=select_distinct(curs,"cohorts","famid","AREA",str(inst))
            print "Added",inst,"there are now",len(families_in_db),"families under consideration"
        families_in_db=set(families_in_db)
        print " There are", len(families_in_db), "families who've been stationed at the sites we're interested in"

```

```
#create the subset of families we haven't seen before
families_to_analyze=[family for family in families_in_db if family not in recorded_families]
print "    ",len(families_to_analyze),"of those haven't been seen before"

#make note that we will have analyzed these folks, so we don't add
#them again when pulling from the other DBs
for family in families_to_analyze: recorded_families.add(family)
print "    ",len(recorded_families),"total families will have been added to the calendar"

#for each family
fams_so_far=0
for family in families_to_analyze:
    cal_curs.execute("begin;")
    #determine members
    family_members=members(curs,family)

    #compile a history for each member
    family_histories={}
    for member in family_members:
        cohort_rows=[cohort_row._make(x) for x in select_a(curs,"cohorts","id",member)]
        if verbose:
            print member
            for row in cohort_rows: print " ",row
        family_histories[member]=(history(cohort_rows))
        if verbose: family_histories[member].print_full_bio()

    #determine if this is a standard family (if not, we can't account
    #for them since there will be multiple or 0 service members)
    sm=standard_family(family_histories)

    #if it's a standard family... otherwise skip it
    if sm:
        #for the lifecycle of each family member, increment the dictionary
        #value for each day/installation/status combination
        for member in family_histories.values():
            #determine what range we want to address.  It will be the inter
            #section of the window and their lifecycle
            days_to_count=set(range(member.arrival(),member.departure()))&set(range(window[0],window[1]+1))

            #address those days
            for day in days_to_count:
                member_installation=member.installation_on_date(day)
                if member_installation in installations:
                    #get the family's sm's status on that day
                    sm_bencat, sm_status, sm_error_flag=family_histories[sm].status_on_date(day)
                    sm_service=family_histories[sm].service[0]

                    #get the family member's status on that day
                    member_bencat, member_status, mem_error_flag=member.status_on_date(day)

```

```
                            #this person's existence on this day is:
                            key_to_update=(member_installation, day, sm_service,
                                member_bencat, sm_status)

                            update_cal_db(cal_curs,key_to_update)
            cal_curs.execute("commit;")
            fams_so_far+=1
            if fams_so_far%10==0: Printer(str(fams_so_far)+" families added to the dict")
        print ""

    print "Adding index for making projections"
    cal_curs.execute("CREATE INDEX IF NOT EXISTS for_arena_seed on calendars (service,bencat
        ,area,day);")
    cal_db=apsw.Connection(new_calendar_database)
    apsw_backup(_cal_db,cal_db)

    #yes, i know... underscore is private... but i want to return the in-memory
    #version and don't have time to rejigger the code above right now.  This
    #doesn't actually need to be private
    return _cal_db

def update_cal_db(cursor,key):
    #calendar[key_to_update]=calendar.get(key_to_update,0)+1
    #command="UPDATE calendars SET persons=persons+1 where "+\
    command="INSERT OR IGNORE INTO calendars values(?,?,?,?,?,?);"+\
            "UPDATE calendars SET persons = persons + 1 where "+\
            "AREA=? "+\
            "AND day=? "+\
            "AND service=? "+\
            "AND BENCAT=? "+\
            "AND status=?;"
    cursor.execute(command,key+(0,)+key)

def gather_rates(folder):
    """
    This function gathers the ACT and DA episode rates from a file, makes
    """
    types=["encs","epis","rvus"]
    services=["A","N","F","M"]
    sm_bencats=["ACT","DA"]#do not include GRD, DGR, IGR, IDG in this list.
                            #they need to be processed differently, since the
                            #relevant rates in those are GRD/IGR, not ACT

    #create an in memory database
    rate_db=apsw.Connection(":memory:")
    r_cur=rate_db.cursor()

    #drop table if exists add columns, including one for each 'type' above
    r_cur.execute("DROP TABLE IF EXISTS rates")
    command="CREATE TABLE rates(service TEXT, BENCAT TEXT, status TEXT"
    for type in types: command=command+", "+type+" TEXT"
    command=command+", PRIMARY KEY (service, BENCAT, status))"
    #print command
    r_cur.execute(command)

    #for each service and each bencat and status, each of which has its own
```

```
    #cpickle file generated
    for service in services:
        for bencat in sm_bencats:
            for type in types:
                #open the related cPickle file for each Active Bencat
                filename="mh_prov_"+type+"-"+service+"-"+bencat+"-rps.cpickle"
                rate_dict=cPickle.load(open(folder+filename))
                #Then add the rates from it to the right spot.  The rates for
                #these BENCATS are in the ACT subdict.  But, they describe the
                #rates for the sm_bencat in question.  So, when inserting to the
                #database, we index them based on their sm_bencat
                print "Reading from",filename
                statuses=rate_dict['ACT'].keys()
                statuses.sort()
                for status in statuses:#<--this is hard coded on purpose
                    #print status, rate_dict['ACT'][status]
                    rate=rate_dict['ACT'][status]
                    #build the command which will insert a row if need be, and
                    #if a row with that combination of bencat, status, service
                    #already exists, then it will just add in our value.
                    command="INSERT or REPLACE INTO rates (service,bencat,status"
                    command=command+","+type
                    for type_not_being_updated in [x for x in types if x!=type]:
                        command=command+","+type_not_being_updated
                    command=command+") values(\'"+service+"\',\'"+bencat+"\',\'"+str(status)+"\'"
                    command=command+",\'"+str(rate)+"\'"
                    for type_not_being_updated in [x for x in types if x!=type]:
                        #command=command+", coalesce((select "+type_not_being_updated+" from rates where BENCAT=\'"+bencat+"\' and service=\'"+service+"\' and status=\'"+str(status)+"\'),'Not inserted')"
                        command=command+",(select "+type_not_being_updated+" from rates where BENCAT=\'"+bencat+"\' and service=\'"+service+"\' and status=\'"+str(status)+"\')"
                    command=command+");"
                    #print command
                    r_cur.execute(command)
                    r_cur.execute("select * from rates where service=\'"+service+"\' and bencat=\'"+bencat+"\' and status=\'"+str(status)+"\';")
                    #print r_cur.fetchall()
    print " "
    return rate_db

def make_projections(cal_db,rate_db,installations,rate_type,location,deterministic=True):
    cc=cal_db.cursor()
    rc=rate_db.cursor()

    #find out what days we're covering
    cc.execute("SELECT distinct day from calendars;")
    days=[x[0] for x in cc.fetchall()]
    days.sort()
    print "From day", min(days),"to",max(days)

    #find all bencats represented in the rate file
    rc.execute("SELECT distinct BENCAT from rates;")
    bencats=[x[0] for x in rc.fetchall()]
```

```
    print "Projecting visits from bencats",bencats

    #find all services represented in the rate file
    rc.execute("SELECT distinct service from rates;")
    services=[x[0] for x in rc.fetchall()]
    print "Projecting visits from services",services

    print "Adding index for making projections"
    cc.execute("CREATE INDEX IF NOT EXISTS for_arena_seed on calendars (service,bencat,area,
        day);")

    #open the file we're using for humans to read (a csv showing demand by inst/day)
    #columns are installation-BENCAT, rows are days
    o=csv.writer(open(location+rate_type+"_projections.csv",'wb'))

    #Make a normal Header
    header=[""]
    for installation in installations:
        for service in services:
            for bencat in bencats:
                header.append(str(installation)+"-"+service+"-"+bencat)
    o.writerow(header)

    #translator is now a function that has k:v pairs preloaded... doesn't need
    #to re-read the csv file every time
    translator=make_translator()

    #Make a header with the installation name
    header=[""]
    for installation in installations:
        for service in services:
            for bencat in bencats:
                header.append(translator(installation)+"-"+service+"-"+bencat)
    o.writerow(header)

    #open the file that will be used to seed ARENA
    #it can't be used directly, but can be opened in excel and saved including
    #the proper 'ranges' that ARENA uses to read data
    seed=None
    if rate_type=='epis': seed=csv.writer(open(location+rate_type+"_arenaseed.csv",'wb'))
    #no headers... that's how ARENA rolls
    #Set number of rows for each day as a static variable.  1000
    patients_per_day=1000

    sn=0
    #CSV Calendar
    for day in days:
        rows_written_for_day=0
        Printer("Adding "+str(day)+" to projections")
        row_to_write=[day]
        for installation in installations:
            for service in services:
                for bencat in bencats:
                    #for each of these, add the summation of rate*pop for each B,P,S
                    total_demand=0
                    cc.execute("SELECT status, persons from calendars where service=\'"+\
```

```
                                service+"\' and bencat=\'"+bencat+"\' and AREA="+\
                                str(installation)+" and day="+str(day)+";")
                    all_current_residents=cc.fetchall()
                    all_current_residents.sort()#not needed, helps with debuging
                    for resident_type in all_current_residents:
                        #add that population times the rate for it's BPS
                        rate_bps=rate(service,bencat,resident_type[0],rate_type,rc)
                        #print "There are",resident_type[1],"in",service,bencat,resident_type[0],"with rate",rate_bps,"at",installation
                        total_demand+=resident_type[1]*rate_bps
                    #create a full row with info for all installations
                    #for the human-readable file
                    row_to_write.append(total_demand)
                    #for the arena file, just add patients as they come
                    if rate_type=='epis':
                        #print "adding" ,total_demand, "on",day
                        patients_to_add=total_demand
                        if not deterministic: patients_to_add=poisson(total_demand)
                        for new_patient in range(int(patients_to_add)):
                            seed.writerow([day,\
                                        str(datetime.date(1970,1,1)+datetime.timedelta(days=day)),\
                                        sn,\
                                        installation,\
                                        bencat])
                            sn+=1
                            rows_written_for_day+=1
        if rate_type=='epis':
            while rows_written_for_day<patients_per_day:
                seed.writerow([day,str(datetime.date(1970,1,1)+datetime.timedelta(days=day)),sn,0,0])
                sn+=1
                rows_written_for_day+=1
        o.writerow(row_to_write)
    print ""

def rate(service, bencat, status,rate_type, cursor):
    """
    returns the rate for a given BPS (rate_type=encs,epis,rvus)
    if the rate isn't in the set, returns 0.  If rate_type=pop, return 1 for
    any person who is actually at the installation (everyone but sms during deps)
    """
    if rate_type=='pop':
        if bencat in ['ACT','GRD'] and status=='during':
            return 0
        else: return 1

    #if the status is an integer (all come in as unicode... ugh) then make sure
    #it's below 3366, if not, subtract 1000 til it is... this lumps all people on
    #their 4th or higher deployment into the rates for the third, since there's
    #a small n on the curves generated for 4th deployments, and it might intro-
    #duce too much "noise"
    try:
        status_i=int(status)
        while status_i>3366: status_i+=-1000
        status=unicode(str(status_i))
```

```
    except ValueError: pass

    cursor.execute("SELECT "+rate_type+" from rates where service=\'"+\
            service+"\' and bencat=\'"+bencat+"\' and status=\'"+\
            status+"\';")
    rate_bps=cursor.fetchall()
    if len(rate_bps)==1:
        return float(rate_bps[0][0])
    else:
        return 0
```

B.2.5 cohort_maker.py

```
#when run, this file makes a subset of the entire data_cube available
#it will output to a folder /sample/ the records for all families
#for which at least one person meets the filter criteria

import csv
import sys
from cohort_handler import cohort_row
import os
from jth_tools import Printer
import time
from multiprocessing import Pool

def filter_a_file(input_tuple):
    family_ids=input_tuple[0]
    file=input_tuple[1]
    out_dir=input_tuple[2]

    print "Writing",file

    o=csv.writer(open(out_dir+file,'wb'))
    a=csv.reader(open(location+file))
    famid_index=None
    last_time=0
    rows_processed=0
    written=0
    for row in a:
        rows_processed+=1
        if famid_index==None:
            famid_index=row.index("famid")
            o.writerow(row)
        else:
            if row[famid_index] in family_ids:
                o.writerow(row)
                written+=1
        if int(time.time())!=last_time:
            last_time=int(time.time())
            #printer.Printer(str(rows_processed)+" rows processed so far. "+str(written)+"
                rows match filter")

def filter_datacube(location,directory,type,filter):
    # This function creates a set of datacube files in the specified subdirectory
    # which contain only the relevant records according to filter type and
    # values
    pool=Pool(processes=4)
    target_dir=location+directory
    os.mkdir(target_dir)

    #first pick family IDS
    family_ids=[]
    next_record=0
    for year in range(2003,2010):
        print "from", year
        a=csv.reader(open(location+"cohort_fy"+str(year)+"_scram.csv"))
        a.next() #skip the header row
```

```
        rows_processed=0
        new_ids=0
        last_time=0
        for row in a:
            #just for informing the user
            rows_processed+=1

            #process the row
            r=cohort_row._make(row)
            if getattr(r,type) in filter:
                new_ids+=1
                family_ids.append(r.famid)

            #print our status occasionally
            if int(time.time())!=last_time:
                last_time=int(time.time())
                Printer(str(rows_processed)+" rows processed so far. "+str(new_ids)+" rows match filter")
        Printer(str(rows_processed)+" rows processed so far. "+str(new_ids)+" rows match filter")
        print ""
    print "Making one set from this list"
    family_ids=set(family_ids)
    print len(family_ids), "IDs total for",type,"filter", filter

    all_files_to_filter=[]
    for year in range(2003,2010):
        #for each file, create a filtered version
        for file in ["cohort_fy"+str(year)+"_scram.csv",\
                     "sadr_fy"+str(year)[2:4]+"_scram.csv",\
                     "sidr_fy"+str(year)[2:4]+"_scram.csv",\
                     "tedni_fy"+str(year)[2:4]+"_scram.csv",\
                     "tedi_fy"+str(year)[2:4]+"_scram.csv",\
                     "pdts_fy"+str(year)[2:4]+"_scram.csv"]:
            #Filter each file in it's own process
            all_files_to_filter.append((family_ids,file,target_dir))

    run_em_all=pool.map(filter_a_file,all_files_to_filter)

if __name__ == '__main__':
    location="/Volumes/Data/DataCube/nas/data/batch_csv/"
    locationPC="Z:/DataCube/nas/data/batch_csv/"
    if os.name=="nt":
        location=locationPC

    if len(sys.argv)==4:
        filter_directory=sys.argv[1]+"/" #first arg is where to put the new files
        filter_type=sys.argv[2] #second arg is what field to filter on
        filter=sys.argv[3] #third field is the acceptable value for the field

        filter_datacube(location,filter_directory,filter_type,filter)
    else: print "didn't understand input"


    #johns_sample_maker_MT.py
#
```

```python
#when run, this file makes a subset of the entire data_cube available
#it will output to a folder /sample/ the records for all families
#for which at least one person meets the filter criteria

import csv
import sys
from cohort_handler import cohort_row
import os
from jth_tools import Printer
import time
from multiprocessing import Pool

def filter_a_file(input_tuple):
    family_ids=input_tuple[0]
    file=input_tuple[1]
    out_dir=input_tuple[2]

    print "Writing",file

    o=csv.writer(open(out_dir+file,'wb'))
    a=csv.reader(open(location+file))
    famid_index=None
    last_time=0
    rows_processed=0
    written=0
    for row in a:
        rows_processed+=1
        if famid_index==None:
            famid_index=row.index("famid")
            o.writerow(row)
        else:
            if row[famid_index] in family_ids:
                o.writerow(row)
                written+=1
        if int(time.time())!=last_time:
            last_time=int(time.time())
            #printer.Printer(str(rows_processed)+" rows processed so far. "+str(written)+" rows match filter")

def filter_datacube(location,directory,type,filter):
    # This function creates a set of datacube files in the specified subdirectory
    # which contain only the relevant records according to filter type and
    # values
    pool=Pool(processes=4)
    target_dir=location+directory
    os.mkdir(target_dir)

    #first pick family IDS
    family_ids=[]
    next_record=0
    for year in range(2003,2010):
        print "from", year
        a=csv.reader(open(location+"cohort_fy"+str(year)+"_scram.csv"))
        a.next() #skip the header row
        rows_processed=0
        new_ids=0
```

```
        last_time=0
        for row in a:
            #just for informing the user
            rows_processed+=1

            #process the row
            r=cohort_row._make(row)
            if getattr(r,type) in filter:
                new_ids+=1
                family_ids.append(r.famid)

            #print our status occasionally
            if int(time.time())!=last_time:
                last_time=int(time.time())
                Printer(str(rows_processed)+" rows processed so far. "+str(new_ids)+" rows match filter")
        Printer(str(rows_processed)+" rows processed so far. "+str(new_ids)+" rows match filter")
        print ""
    print "Making one set from this list"
    family_ids=set(family_ids)
    print len(family_ids), "IDs total for",type,"filter", filter

    all_files_to_filter=[]
    for year in range(2003,2010):
        #for each file, create a filtered version
        for file in ["cohort_fy"+str(year)+"_scram.csv",\
                     "sadr_fy"+str(year)[2:4]+"_scram.csv",\
                     "sidr_fy"+str(year)[2:4]+"_scram.csv",\
                     "tedni_fy"+str(year)[2:4]+"_scram.csv",\
                     "tedi_fy"+str(year)[2:4]+"_scram.csv",\
                     "pdts_fy"+str(year)[2:4]+"_scram.csv"]:
            #Filter each file in it's own process
            all_files_to_filter.append((family_ids,file,target_dir))

    run_em_all=pool.map(filter_a_file,all_files_to_filter)

if __name__ == '__main__':
    location="/Volumes/Data/DataCube/nas/data/batch_csv/"
    locationPC="Z:/DataCube/nas/data/batch_csv/"
    if os.name=="nt":
        location=locationPC

    if len(sys.argv)==4:
        filter_directory=sys.argv[1]+"/" #first arg is where to put the new files
        filter_type=sys.argv[2] #second arg is what field to filter on
        filter=sys.argv[3] #third field is the acceptable value for the field

        filter_datacube(location,filter_directory,filter_type,filter)
    else: print "didn't understand input"
```

B.2.6 db_curve_maker.py

```
"""
This is a ground up rewrite of the original db_curve_maker.
The first version died and untimely death due to a file corruption
after a hard disconnect of the NAS (fml)

This file, when run as a script, calculates the care seeking rates for each
of the beneficiary categories listed in each of the services listed.

It does this once in encounters, once for new episodes of care (arrivals),
and once for RVUs

"""



#python library imports
import os, sys
from collections import namedtuple
from optparse import OptionParser

#special libraries needed
import apsw #Another Python SQLite Wrapper. Needed to Load DB to memory.

#from other files
from jth_tools import jth_apsw_tracer, absorb, Printer
from new_history_handler import history
from merge_to_rates import merge
from datehandler import sasd2ed
from db_tools import *


def get_status_on_dates(deployer,date_list):
    statuses_on_dates={}
    for date in date_list:
        bencat,status,error_flag=deployer.status_on_date(date)
        if error_flag and verbose:
            #a small number of encounters fall outside of the cohort records for the service
                 member
            #this is true even for service member encounters (>.1% of them)
            #and for encounters belonging to Dependents of Surviors (DS)
            print "Cant correlate encounter on",date,"with"
            deployer.print_full_bio()
        elif error_flag: pass
        else:
            if bencat not in statuses_on_dates: statuses_on_dates[bencat]={}
            if status not in statuses_on_dates[bencat]: statuses_on_dates[bencat][status]=0
            statuses_on_dates[bencat][status]+=1
    if verbose:
        print "The encounters on dates",date_list
        print "for the servicemember"
        deployer.print_full_bio()
        print "return statuses", statuses_on_dates
    return statuses_on_dates

```

```
def get_status_on_dates_rvus(deployer,date_rvus_list):
    statuses_on_dates={}
    for date in date_rvus_list:
        bencat,status,error_flag=deployer.status_on_date(date[0])
        if error_flag and verbose:
            #a small number of encounters fall outside of the cohort records for the service member
            #this is true even for service member encounters (>.1% of them)
            #and for encounters belonging to Dependents of Surviors (DS)
            print "Cant correlate encounter on",date,"with"
            deployer.print_full_bio()
        elif error_flag: pass
        else:
            if bencat not in statuses_on_dates: statuses_on_dates[bencat]={}
            if status not in statuses_on_dates[bencat]: statuses_on_dates[bencat][status]=0
            statuses_on_dates[bencat][status]+=date[1]
    if verbose:
        print "The encounters on dates",date_rvus_list
        print "for the servicemember"
        deployer.print_full_bio()
        print "return statuses", statuses_on_dates
    return statuses_on_dates

def get_encounters_ps(all_encounters, deployer):
    """
    returns a dictionary representing the incidence of encounters
    in treated_person's history correlated with their position in the deployer's
    position in the deployment cycle.

    both arguments should be history objects
    """
    #get all the dates of encounters
    all_enc_dates=[sasd2ed(int(x.begdate)) for x in all_encounters]
    all_enc_dates.sort()
    if verbose: print "full set of encounters ", all_enc_dates

    #if no encounters are passed to it, return an empty dict
    if len(all_enc_dates)==0: return {}

    return get_status_on_dates(deployer, all_enc_dates)

def get_rvus_ps(all_encounters, deployer):
    """
    returns a dictionary representing the incidence of encounters
    in treated_person's history correlated with their position in the deployer's
    position in the deployment cycle.

    both arguments should be history objects
    """
    #get all the dates of encounters
    all_enc_dates_rvus=[(sasd2ed(int(x.begdate)),0.0 if x.wgtd_work == "" else float(x.wgtd_work)) for x in all_encounters]
    all_enc_dates_rvus.sort()
    if verbose: print "full set of encounters ", all_enc_dates_rvus

    #if no encounters are passed to it, return an empty dict
```

```
    if len(all_enc_dates_rvus)==0: return {}

    return get_status_on_dates_rvus(deployer, all_enc_dates_rvus)

def get_episodes_ps(all_encounters, deployer):
    """
    returns a dictionary representing the incidence of the beginning of episodes
    in treated_person's history correlated with their position in the deployer's
    position in the deployment cycle.

    both arguments should be history objects
    """
    weeks_gap_to_end_episode=8
    days_gap_to_end_episode=weeks_gap_to_end_episode*7

    #get all the dates of encounters
    all_enc_dates=[sasd2ed(int(x.begdate)) for x in all_encounters]
    all_enc_dates.sort()
    if verbose: print "from original set of encounters ", all_enc_dates

    #if no encounters are passed to it, return an empty dict
    if len(all_enc_dates)==0: return {}

    #filter to only include ones that represent the start of episodes
    last_encounter=None
    episode_starts=[]
    for enc_date in all_enc_dates:
        if last_encounter==None or enc_date>last_encounter+days_gap_to_end_episode:
            episode_starts.append(enc_date)
        last_encounter=enc_date

    if verbose: print "only",episode_starts,"were the start of new episodes"

    return get_status_on_dates(deployer, episode_starts)

def meets_filters(histories, filters):
    """
    Return a subset of the original dictionary containing all family members who
    spend at least some time meeting each of the filters.

    In addition, family members must have a coherent history, defined by the
    history.unclear() method.  Unclear histories are left out.
    """

    meets_all={}
    for member in histories:
        meets=True
        for filter in filters:
            #so long as the person meets that filter at any time, we're okay
            if len(set(getattr(histories[member],filter[0]))&set(filter[1])): pass
            #otherwise, that person is excluded
            else:
                if verbose: print member,"was rejected because they didn't pass",filter[0]," filter"
                meets=False

```

```
            if meets: meets_all[member]=histories[member]

    if verbose:
        print "original set was",histories.keys()
        print "set meeting filters was", meets_all.keys()
    return meets_all


def make_rates(filter_set, location, to_mem=True, enc_processor=get_episodes_ps, file_leader
    ="mh_prov"):
    """
    This function makes a csv file and a cPickle file that contain the
    rate information for a set of beneficiaries in standard families
    who conform to the filter set
    """

    #establish connections to cohort database
    _cohort_db=apsw.Connection(location+"cohorts.db")
    #establish connection to encounter database
    _enc_db=apsw.Connection(location+"mh_prov_encounters.db")

    #if to_mem, load the databases into RAM
    cohort_db=None
    enc_db=None
    if to_mem:
        cohort_db=apsw.Connection(":memory:")
        with cohort_db.backup("main",_cohort_db,"main") as backup:
            print "loading " +location+"cohorts.db"
            while not backup.done:
                backup.step(10000)
                Printer(str(100.0*backup.remaining/backup.pagecount)+" percent remaining")

        enc_db=apsw.Connection(":memory:")
        with enc_db.backup("main",_enc_db,"main") as backup:
            print "loading "+location+"mh_prov_encounters.db"
            while not backup.done:
                backup.step(10000)
                Printer(str(100.0*backup.remaining/backup.pagecount)+" percent remaining")
            print(" Complete")
    else:
        cohort_db=_cohort_db
        #enc_db=_enc_db



    #get cursors
    coh_curs=cohort_db.cursor()
    enc_curs=enc_db.cursor()

    #create a list of all family ids in the cohort database
    families=select_distinct(coh_curs,"cohorts","famid")
    print "Analyzing", len(families),"families"
    fams_in_enc_db=select_distinct(enc_curs,"encounters","famid")
    print "There are", len(fams_in_enc_db),"families with mh_prov visits in the database"
    individuals_in_enc_db=select_distinct(enc_curs,"encounters","id")
    print "There are", len(individuals_in_enc_db),"people with mh_prov visits in the
```

```
        database"

    #get the cohort columns
    coh_columns=get_columns(coh_curs, "cohorts")
    #define a named tuple for the format of the cohort database
    cohort_row=namedtuple("cohortdbrow",coh_columns)

    #get the encounter columns
    enc_columns=get_columns(enc_curs, "encounters")
    #define a named tuple for the format of the encounter database
    encounter_row=namedtuple("encounterdbrow",enc_columns)

    #start with empty dictionaries for the encounters and days lived in each status
    events_ps={}
    dps={}
    #start counters for # of entities processed for status updates and summary
    families_considered=0
    persons_considered=0
    families_processed=0
    persons_processed=0

    #for each of those families
    for famid in families:
        #find the id of all family members
        family_members=members(coh_curs,famid)
        if verbose: print "family",famid,"has",len(family_members),"members"

        #create a "history" object for each family member
        family_histories={}
        for member in family_members:
            cohort_rows=[cohort_row._make(x) for x in select_a(coh_curs,"cohorts","id",
                member)]
            if verbose:
                print member
                for row in cohort_rows: print " ",row
            family_histories[member]=(history(cohort_rows))
            if verbose: family_histories[member].print_full_bio()

        #if this is a "standard family" then analyze it. Otherwise, skip it
        sm = standard_family(family_histories) #returns false for non-standard families, or
            the id of the servicemember
        if sm:
            if verbose: print "this is a standard family"
            #for each family member with the relevant characteristics from filter_set
            for member in meets_filters(family_histories, filter_set):
                #get the relevant encounters.
                if verbose: print "adding the relevant encounters for",member,"correlated to
                     their sm sponsors deployment cylce"
                all_encounters=[encounter_row._make(x) for x in select_a(enc_curs,'
                    encounters','id',member)]
                p_events_ps=enc_processor(all_encounters,family_histories[sm])

                #user may specify another function (eg one that returns only dxs).
                #correlate all encounters to a position in the sponsor's
                #deployment cycle

```

```
                #find out how many days the sponsor SM spent in each status during
                #that family member's lifecycle
                a=family_histories[member].arrival()
                d=family_histories[member].departure()
                if verbose: print "====>Adding the days the sponsor spent in service between dates",a,"and",d
                p_dps=family_histories[sm].personal_dps(a,d)
                if verbose: print "====>",p_dps
                #add encounters per status and days per status to the running tally
                events_ps=absorb(events_ps,p_events_ps)

                dps=absorb(dps,p_dps)

                persons_processed+=1
            families_processed+=1
        if families_considered%1000==0: print families_considered,"Families Considered"
        persons_considered+=len(family_members)
        families_considered+=1

    print "  Families Considered:",families_considered
    print "  Families Processed: ",families_processed
    print "  Persons Considered: ",persons_considered
    print "  Persons Processed:  ",persons_processed

    filter_string="-"+"-".join(["-".join(filter[1]) for filter in filter_set])+"-"
    if verbose: print "writing to a file with",filter_string,"in the name"

    return merge(dps, events_ps, location, '', True, file_leader+filter_string)

if __name__ == '__main__':
    parser=OptionParser()
    parser.add_option("-v","--verbose",action="store_true",dest="verbose",default=False)
    parser.add_option("-t","--temp",action="store_true",dest="use_temp_dir",default=False)
    parser.add_option("-s","--subfolder",action="store",type='string',dest="sample_dir",default="")
    (options,args)=parser.parse_args()

    sample_dir=options.sample_dir
    verbose=options.verbose
    use_temp_dir=options.use_temp_dir

    location=""
    if use_temp_dir:
        location = ('/Users/johnhess/Desktop/DataCubeTemp/' + sample_dir+"/")
    else:
        location = ('/Volumes/Data/DataCube/nas/data/batch_csv/' + sample_dir+"/")

    if verbose: print "location is",location

    services=[]

    #full service cohorts
    services=services+[  ("FullArmyData/",["A"]),\
                ("FullNavyData/",["N"]),\
                ("FullUSMCData/",["M"]),\
                ("FullUSAFData/",["F"])]
```

```
    for ben in ["ACT","DA"]:
        for service in services:
            sub_location=location+service[0]
            filter_set=[]
            filter_set.append(("service",service[1]))
            filter_set.append(("BENCAT",[ben]))
            make_rates(filter_set, sub_location, to_mem=True, enc_processor=get_encounters_ps, file_leader="mh_prov_encs")
            make_rates(filter_set, sub_location, to_mem=True, enc_processor=get_episodes_ps, file_leader="mh_prov_epis")
```

B.2.7 cohort_handler.py

```
#contains a subclass for reading rows out of cohort files
#this is a subclass of namedtuple and lets us do all this without the cdict crap
#from my old files

from collections import namedtuple

class cohort_row(namedtuple("cohort_row","BENCAT MARITAL PARC BEGELG_DT FY RANK PATDOB
    PATSEX RACE ETHNIC RET_TYPE AREA service ENDELG_DT BENCAT_CHG MARITAL_CHG RANK_CHG
    AREA_CHG BENCAT_DT MARITAL_DT RANK_DT AREA_DT NUM_MOVE ELG_MNTHS ENR_MNTHS JSM_FLAG
    numdep depdays startdep enddep dc_adm dc_adm_mh dc_adm_anymh dc_eradm dc_days dc_days_mh
     dc_days_anymh dc_rwp dc_rwp_mh dc_rwp_anymh dc_msrwp dc_msrwp_mh dc_msrwp_anymh
    dc_ipcost dc_ipcost_mh dc_ipcost_anymh dc_enc dc_enc_mh dc_enc_anymh dc_enc_priv
    dc_enc_npriv dc_erenc dc_erenc_mh dc_erenc_anymh dc_rvu dc_rvu_mh dc_rvu_anymh dc_opcost
     dc_opcost_mh dc_opcost_anymh pc_enc pc_enc_mh pc_enc_anymh pc_enc_priv pc_enc_npriv
    pc_erenc pc_erenc_mh pc_erenc_anymh pc_rvu pc_rvu_mh pc_rvu_anymh pc_allow pc_allow_mh
    pc_allow_anymh pc_paid pc_paid_mh pc_paid_anymh pc_ac_adm pc_ac_adm_mh pc_ac_adm_anymh
    pc_ac_days pc_ac_days_mh pc_ac_days_anymh pc_ac_eradm pc_ac_paid pc_ac_paid_mh
    pc_ac_paid_anymh pc_ac_allow pc_ac_allow_mh pc_ac_allow_anymh pc_ac_rwp pc_ac_rwp_mh
    pc_ac_rwp_anymh pc_ac_msrwp pc_ac_msrwp_mh pc_ac_msrwp_anymh pc_nac_adm pc_nac_adm_mh
    pc_nac_adm_anymh pc_nac_days pc_nac_days_mh pc_nac_days_anymh pc_nac_paid pc_nac_paid_mh
     pc_nac_paid_anymh pc_nac_allow pc_nac_allow_mh pc_nac_allow_anymh pc_ac_profpaid
    pc_ac_profallow pc_ac_profrvu pc_ac_profpaid_mh pc_ac_profallow_mh pc_ac_profrvu_mh
    pc_ac_profpaid_anymh pc_ac_profallow_anymh pc_ac_profrvu_anymh d30equiv d30equiv_mh
    d30equiv_nar script_mtf_tot script_mtf_mh script_mtf_nar script_retail_tot
    script_retail_mh script_retail_nar script_mail_tot script_mail_mh script_mail_nar
    rxcost_mtf_tot rxcost_mtf_mh rxcost_mtf_nar rxpaid_retail_tot rxpaid_retail_mh
    rxpaid_retail_nar rxallow_retail_tot rxallow_retail_mh rxallow_retail_nar
    rxpaid_mail_tot rxpaid_mail_mh rxpaid_mail_nar rxallow_mail_tot rxallow_mail_mh
    rxallow_mail_nar hcc mh_flag subabuse_flag mh_history ahrq_adjust ahrq_anxiety
    ahrq_atten ahrq_devel ahrq_child ahrq_impulse ahrq_mood ahrq_person ahrq_schizo
    ahrq_alcohol ahrq_subabuse ahrq_suicide ahrq_miscmh ahrq_screen iiw_ptsd iiw_tbi
    iiw_ampute iiw_burn iiw_spinal iiw_shrapnel iiw_fracture iiw_blind iiw_sigtrauma cohort
    famid scram_edipn")):
    pass
```

B.2.8 date_handler.py

```
#this file contains a set of used and disused functions and classes for
#translating between different types of dates:
#   sd=string date ("MM/DD/YY") and the like, used for startdep and endep in the
#                      datafiles provided
#   ed=epoch date, days since 1 JAN 1970
#   sasd=SAS date, days since 1 JAN 1960
#   dt=python date
#
#In hindsight, converting things into epoch dates is ill advised (the integer SAS
#date format is just as efficient).  The decision was made when working only
#with string dates, which needed a convenient and quick integer formulation.

import datetime

class sd2ed():
    def __init__(self,sdate):
        month=""
        day=""
        year=""
        slashes=0
        for letter in sdate:
            if letter=="/":
                slashes+=1
            elif slashes==0:
                month=month+letter
            elif slashes==1:
                day=day+letter
            elif slashes==2:
                year=year+letter

        edate=datetime.date(int(year),int(month),int(day))

        self.dif=edate-datetime.date(1970,1,1)

    def ed(self):
        return self.dif.days

def sasd2ed(sasd):
    dif6070=datetime.date(1970,1,1)-datetime.date(1960,1,1)
    edate=sasd-dif6070.days
    return edate

class ed2sd():
    def __init__(self,ed):
        self.ed_as_date=datetime.date(1970,1,1)+datetime.timedelta(days=ed)

    def sd(self):
        return self.ed_as_date

class absorb_date():
    def __init__(self,ed_in=False,sd_in=False,sasd_in=False,dt_in=False):
        #First, process a sas date, so that we can go sas-->epoch-->actual date
        if sasd_in != False:
            dif6070=datetime.date(1970,1,1)-datetime.date(1960,1,1)
```

```
            ed_in=sasd_in-dif6070.days
        if ed_in != False: self.date_in=datetime.date(1970,1,1)+datetime.timedelta(days=
            ed_in)
        if sd_in != False:
            month=""
            day=""
            year=""
            slashes=0
            for letter in sd_in:
                if letter=="/":
                    slashes+=1
                elif slashes==0:
                    month=month+letter
                elif slashes==1:
                    day=day+letter
                elif slashes==2:
                    year=year+letter
            year=int(year)
            if year<15 and year>0:
                year=year+2000
            self.date_in=datetime.date(year,int(month),int(day))
        if dt_in != False: self.date_in=dt_in

    def sd(self):
        return str(self.date_in)
    def ed(self):
        epoch_days=self.date_in-datetime.date(1970,1,1)
        return epoch_days.days
    def sasd(self):
        sas_days=self.date_in-datetime.date(1960,1,1)
        return sas_days.days
    def dt(self):
        return self.date_in
```

B.2.9 db_tools.py

```
"""
Contains a set of functions that are handy for doing database work via apsw
"""
import apsw

verbose=False

def select_distinct(cursor, table_name, field_name, filter_field=None, filter_field_value=
    None):
    #returns the list representing all distinct field_name values in table
    #with table_name in the database behind cursor.
    cmd=""
    if filter_field:
        cmd="SELECT DISTINCT "+field_name+" FROM "+table_name+" WHERE "+filter_field+" = \'"
            +filter_field_value+"\';"
    else:
        cmd="SELECT DISTINCT "+field_name+" FROM "+table_name+";"
    if verbose: print cmd
    cursor.execute(cmd)
    distinct_vals = cursor.fetchall()
    distinct_vals = [x[0] for x in distinct_vals] #get only the first value of each of the
        tuples, since each tuple contains one value

    return distinct_vals

def select_a(cursor, table_name, filtered_field, filtered_field_value):
    #returns the whole row where (i.e. SELECT * WHERE filtered_field=filtered_field_value)
    cmd="SELECT * FROM "+table_name+" WHERE "+filtered_field+" = \'"+str(
        filtered_field_value)+"\';"
    if verbose: print cmd
    cursor.execute(cmd)
    return(cursor.fetchall())

def get_columns(cursor, table_name):
    #returns the columns for a table
    cmd="SELECT * FROM "+table_name+" limit 1;"
    cursor.execute(cmd)
    columns=[x[0] for x in cursor.getdescription()]  #just the name, not type
    if verbose: print columns
    return columns

def members(coh_curs,famid):
    return select_distinct(coh_curs,"cohorts","id","famid",famid)

def standard_family(histories):
    """
    This function return the id of the sole service member (defined in sm_bencats)
    or False if the family has zero or more than one service member.
    """
    sm_bencats=["ACT","GRD","IGR","RET"]
    dep_bencats=["DA","DR","DS","IDG","DGR"]

    sms=0
    sm=""
```

```
    for member in histories:
        if verbose: print "person had history of being",histories[member].BENCAT
        if len(set(sm_bencats)&set(histories[member].BENCAT))>0 and len(set(dep_bencats)&set
            (histories[member].BENCAT))==0:
            sms+=1
            sm=member
        if histories[member].unclear():
            if verbose: print member,"Caused the family to be rejected because of an unclear
                 history"
            return False

    if sms==1:
        return sm
    else: return False
```

B.2.10 jth_tools.py

```
#This file contains a set of re-usable tools that are shared across a few
#projects

from collections import namedtuple
import copy
from sys import stdout
import apsw
import csv,os

def aggregate(i1,i2):
    """
    this recursive function aggregates two similar entities
    its intended use is to merge two dictionaries

    returns a dictionary containing all entries that are present in any dict
    if the same unique entry is present in both dicts, then the value is an
    aggregate of those entries.  If the values are integers/floats then the
    aggregate is the sum of values.  If the values are lists, then the
    lists are merged

    At no point in the nested dictionaries should the types of various
    levels be mismatched (ie a dictionary of dicts and a dictionary of lists
    cannot be added)

    >>> aggregate({},{})
    {}
    >>> aggregate({"a":5},{"b":5,"a":1})
    {'a': 6, 'b': 5}
    >>> a={'outer_dict':{'inner_int': 10,'inner_list':[50,50,60]}, 'outer_int': 5, 'outer_list':[1,2,3]}
    >>> b={'outer_dict':{'inner_int':1,'inner_list':[70]}, 'outer_int': 1, 'outer_list':[4,5,6], 'something else': 5}
    >>> aggregate(a,b)["something else"]
    5
    >>> aggregate(a,b)["outer_dict"]["inner_int"]
    11
    >>> aggregate(a,b)["outer_dict"]["inner_list"]
    [50, 50, 60, 70]
    >>> aggregate(a,b)["outer_list"]
    [1, 2, 3, 4, 5, 6]
    >>> aggregate("ham", "eggs")
    Unknown types.  Lists, Dictionaries, Ints and Floats Accepted
    False

    """

    a=False

    if type(i1) != type(i2):
        print "Mismatched types"
        pass
    elif (type(i1)==bool and i1==False) or (type(i2)==bool and i2==False):
        #pass failures to add back up the recursive chain
        pass
```

```
    elif type(i1) == dict:
        a={}
        for key in set(set(i1.keys())|set(i2.keys())):
            if key in i1 and key in i2:
                a[key]=aggregate(i1[key],i2[key])
            elif key in i1:
                a[key]=i1[key]
            elif key in i2:
                a[key]=i2[key]
    elif type(i1) in [int,float,list]:
        a = i1+i2
    else:
        print "Unknown types.  Lists, Dictionaries, Ints and Floats Accepted"
    return a

def absorb(i1,i2):
    """
    this recursive function aggregates two similar entities
    its intended use is to merge two dictionaries

    returns a dictionary containing all entries that are present in any dict
    if the same unique entry is present in both dicts, then the value is an
    aggregate of those entries.  If the values are integers/floats then the
    aggregate is the sum of values.  If the values are lists, then the
    lists are merged

    At no point in the nested dictionaries should the types of various
    levels be mismatched (ie a dictionary of dicts and a dictionary of lists
    cannot be added)

    >>> absorb({},{})
    {}
    >>> absorb({"a":5},{"b":5,"a":1})
    {'a': 6, 'b': 5}
    >>> a={'outer_dict':{'inner_int': 10,'inner_list':[50,50,60]}, 'outer_int': 5, 'outer_list':[1,2,3]}
    >>> b={'outer_dict':{'inner_int':1,'inner_list':[70]}, 'outer_int': 1, 'outer_list':[4,5,6], 'something else': 5}
    >>> absorb(a,b)["something else"]
    5
    >>> a={'outer_dict':{'inner_int': 10,'inner_list':[50,50,60]}, 'outer_int': 5, 'outer_list':[1,2,3]}
    >>> absorb(a,b)["outer_dict"]["inner_int"]
    11
    >>> a={'outer_dict':{'inner_int': 10,'inner_list':[50,50,60]}, 'outer_int': 5, 'outer_list':[1,2,3]}
    >>> absorb(a,b)["outer_dict"]["inner_list"]
    [50, 50, 60, 70]
    >>> a={'outer_dict':{'inner_int': 10,'inner_list':[50,50,60]}, 'outer_int': 5, 'outer_list':[1,2,3]}
    >>> absorb(a,b)["outer_list"]
    [1, 2, 3, 4, 5, 6]
    >>>

    """

```

```
    if type(i1) == dict:
        for key in set(i2.keys()):
            if key in i1:
                i1[key]=absorb(i1[key],i2[key])
            elif key in i2:
                i1[key]=i2[key]
    else:
        i1 = i1+i2
    return i1

def jth_apsw_tracer(cursor,row):
    field_names=[x[0] for x in cursor.getdescription()]
    return dict(zip(field_names,row))

class Printer():
    #dynamically print to one line
    def __init__(self,data):
        if type(data) is str:
            stdout.write("\r\x1b"+"  " + data.__str__())
            stdout.flush()
        elif type(data) is dict:
            if "specs" in data:
                print "Specs:",data["specs"]
                for installation in data.iterkeys():
                    if installation<>"specs" and data[installation]["CareCounter"]<>{}:
                        print " %4s %15s %30s" %(installation,data[installation]["InstName"][0:12],data[installation]["DMISName"][0:25])
                        for day in data[installation]["CareCounter"].iterkeys():
                            print "  " ,day, data[installation]["CareCounter"][day]
            else: print "No entry called 'specs'"

def apsw_backup(from_db=None, to_db=None):
    with to_db.backup("main",from_db,"main") as b:
        while not b.done:
            b.step(10000)
            Printer("Backing up database: "+str(100.0*b.remaining/b.pagecount)+" percent remaining")
        print ""


#returns a function that can be used to translate area numbers to names
def make_translator():
    location="/Volumes/Data/DataCube/nas/data/batch_csv/"
    locationPC="Z:/DataCube/nas/data/batch_csv/"
    if os.name=="nt":
        location=locationPC

    kv_pairs={}
    handle=open(location+"installations.csv")
    a=csv.reader(handle)
    a.next()
    for row in a:
        kv_pairs[int(row[0])]=row[7]

    def translate(area):
        if type(area) is int: return kv_pairs[area]
```

```
        else: return area
    return translate

class memoize(object):
    def __init__(self,func):
        self.func=func
        self.cache={}

    def __call__(self,*args):
        try:
            return self.cache[args]
        except KeyError:
            value=self.func(*args)
            self.cache[args]=value
            return value
        except TypeError:
            #can't cache based on the args provided (e.g. a list)
            #just return the value
            return self.func(*args)

    def __repr__(self):
        return self.func.__doc__


if __name__ == '__main__':
    import doctest
    doctest.testmod()
```

B.2.11 merge_to_rates.py

```
#takes days per status and encounters per status
#writes a csv file of rates per, days per and encs per status
import csv
import cPickle

all_bencats=["ACT","RET","GRD","IGR","DA","DR","DS","DGR","IDG","OTH","Z"]
dep_bencats=["ACT","GRD"]

def merge(dps,eps,l,f,save,prefix=""):
    o=csv.writer(open(l+f+prefix+"rates.csv",'wb'))
    all_statuses=[]
    bencat_order=[]
    for BENCAT in set(dps.keys())|set(eps.keys()):
        if BENCAT not in eps: eps[BENCAT]={}
        if BENCAT not in dps: dps[BENCAT]={}
        for status in set(dps[BENCAT].keys())|set(eps[BENCAT].keys()):
            all_statuses.append(status)
        bencat_order.append(BENCAT)
    all_statuses=list(set(all_statuses))

    header=[""]
    second=["Status"]
    for bencat in bencat_order:
        header.append(bencat)
        second.append("Days")
        header.append(bencat)
        second.append("Encounters")
        header.append(bencat)
        second.append("Encs/Day")

    o.writerow(header)
    o.writerow(second)

    all_statuses.sort()
    all_statuses.reverse()

    for status in [textual for textual in all_statuses if type(textual) is str]:
        row_to_write=[status]
        for BENCAT in bencat_order:
            row_to_write.append(dps[BENCAT].get(status,""))
            if BENCAT not in eps: eps[BENCAT]={}
            row_to_write.append(eps[BENCAT].get(status,""))
            if dps[BENCAT].get(status,False):
                row_to_write.append(1.0*eps[BENCAT].get(status,0)/dps[BENCAT][status])
            else: row_to_write.append("")
        o.writerow(row_to_write)

    all_statuses.sort()

    for status in [numerical for numerical in all_statuses if type(numerical) is int]:
        row_to_write=[status]
        for BENCAT in bencat_order:
            row_to_write.append(dps[BENCAT].get(status,""))
            if BENCAT not in eps: eps[BENCAT]={}
```

```
            row_to_write.append(eps[BENCAT].get(status,""))
            if dps[BENCAT].get(status,False):
                row_to_write.append(1.0*eps[BENCAT].get(status,0)/dps[BENCAT][status])
            else: row_to_write.append("")
        o.writerow(row_to_write)

    if save: return rps_save(dps,eps,l,f,prefix)

def rps_save(dps,eps,l,f,prefix):
    #This function will save a pickled dictionary that records the encounter (or dx)
    #rate for each bencat in each status
    rps={}
    for BENCAT in set(all_bencats)|set(dps.keys())|set(eps.keys()):
        rps[BENCAT]={}
        #deal with missing eps and entries
        if BENCAT not in eps: eps[BENCAT]={}
        if BENCAT not in dps: dps[BENCAT]={}

        statuses_to_record=["pre","post","during"]
        if BENCAT in dep_bencats:
            #then we're writing not just a pre/post, but also daily deltas
            for deployment in [1,2,3]:
                for delta in range(0,367):
                    statuses_to_record.append(1000*deployment+delta)
        #write to rps for each relevant bencat/status combo
        for status in statuses_to_record:
            if status in dps.get(BENCAT,[]):
                if dps[BENCAT][status]==0 and eps[BENCAT].get(status,0)==0: rps[BENCAT][status]=0 #for the case when no days are spent in a status
                elif dps[BENCAT][status]==0: print "no days in:",BENCAT,status,", but",eps[BENCAT][status],"encounters"
                else: rps[BENCAT][status]=1.0*eps[BENCAT].get(status,0)/dps[BENCAT][status]
            else:
                rps[BENCAT][status]=0

    #now, write it to a file
    print "Making intermediate data file rps.cpickle"
    o=open(l+f+prefix+"rps.cpickle",'wb')
    dumper=cPickle.Pickler(o)
    dumper.fast=True
    dumper.dump(rps)
    print "Finished dumping file"

    return rps
```

B.2.12 new_history_handler.py

```
#This file should allow importing namedtuple type history
from collections import namedtuple
from datehandler import sasd2ed, sd2ed, absorb_date
from cohort_handler import cohort_row


import datetime
import csv
import os
import cPickle
from pprint import pprint
from timeline_parser import overlap

verbose=True
test=False

all_bencats=["ACT","RET","GRD","IGR","DA","DR","DS","DGR","IDG","OTH","Z"]
dep_bencats=["ACT","GRD"]

class history():
    # Contains lifecycle information for a beneficiary based on cohort data
    def __init__(self, cohort_rows):
        """
        Accepts two arguments from the outside: ordered rows from the cohort file
        which are indexable by field name (e.g. named tuples) and all
        encounters which are relevant (e.g. all mh_prov rows)
        """

        self.years=[]
        self.TimeLine=[]
        self.BENCAT=[]
        self.service=[]
        self.PARC=[]
        for row in cohort_rows:
            self.years.append(row.FY)
            if row.service not in self.service: self.service.append(row.service)
            if row.BENCAT not in self.BENCAT: self.BENCAT.append(row.BENCAT)
            if row.PARC not in self.PARC and row.PARC!="": self. PARC.append(row.PARC)
            self.update(row)
            self.id=row.id
            self.famid=row.famid

    def valid(self):

        #this checks to see if the timeline makes any sense.
        #it repairs things it can, and returns false if the record is FUBAR
        #because there may be many things wrong, it makes corrections and
        #recursively calls itself on the newly modified version

        self.TimeLine.sort()
        num_entries=len(self.TimeLine)

        #see if we need to eliminate an eor that isn't the last thing in the
        #record. Note that this does not ensure that the timeline ends with
```

```
#an eor
for entry in range(0,num_entries-1):
    if self.TimeLine[entry][1]=="eor":
        del self.TimeLine[entry]
        return self.valid()

#if there are any overlapping deployments, merge them into one.
#if there are multiple returns or departures and a single of the other
#then delete the one in the middle (again, merge them)

#check for multiple returns from same deployment
hunt=False
for entry in range(num_entries-1,-1,-1):
    #go in reverse
    if self.TimeLine[entry][1]=="d": hunt=False
    elif self.TimeLine[entry][1]=="r" and hunt:
        #print "deleting improper ret from"
        #self.print_timeline(verbose=True)
        del self.TimeLine[entry]
        return self.valid()
    elif self.TimeLine[entry][1]=="r": hunt=True

#check for multiple departures for the same deployment
hunt=False
for entry in range(0,num_entries):
    if self.TimeLine[entry][1]=="r": hunt=False
    elif self.TimeLine[entry][1]== "d" and hunt:
        #print "deleting improper dep from"
        #self.print_timeline(verbose=True)
        del self.TimeLine[entry]
        return self.valid()
    elif self.TimeLine[entry][1]=="d": hunt=True

#finally, check for an irregular deployment pattern, print it.
dr="r"
for entry in self.TimeLine:
    #a normal deployment
    if entry[1]=="d" and dr=="r": dr="d"
    #a normal return
    elif entry[1] =="r" and dr=="d": dr="r"
    elif type(entry[1]) is int: pass
    elif entry[1] in all_bencats: pass
    elif entry[1] == "eor": pass
    elif entry[1] == "UNKNOWN":pass
    else:
        self.print_full_bio()
        print "Invalid history"
        return False

#if the person deploys, make sure that they're in the right BENCAT
last_known_BENCAT=0
for entry in self.TimeLine:
    if entry[1]=="d" and last_known_BENCAT not in dep_bencats:
        #print "WARNING:", last_known_BENCAT,"has a deployment"
        #self.print_full_bio()
        return False
```

```
            elif entry[1] in all_bencats: last_known_BENCAT=entry[1]

        return True

    def update(self,cr):
        """
        This method updates the history to reflect an entry from the cohort file

        To do so, it needs data from several columns.  These are passed to it
        in an object cr which is indexed by column name (e.g. a named tuple)
        Required indices:
            FY: Integer. The year the cohort row represents
            BEGELG_DT: Integer. The record's entry representing the start of that person's
                eligibility
            ENDELG_DT: Integer. Expected end of eligibility
            startdep:
            enddep: Strings. These two represent a recent deployment.
            BENCAT_DT: Integer. If there's a change in bencat, this is when
            AREA_DT: Integer. If there's a change in AREA, this is when
            AREA: Integer.  DMIS code for that person's enrollment
            BENCAT: String.  That persons beneficiary category
            BENCAT_CHG: Integer.  Describes change with retirement, etc.

        With the exception of deployment dates, all dates in our source files
        are integer=num of days since Jan 1 1960.  All dates are converted to
        epoch dated integers (days since Jan 1 1970)
        """

        #Parse the input
        #Calculate the start and end of this FY as epoch dated integers
        START_FY=absorb_date(dt_in=datetime.date(int(cr.FY)-1,10,1))
        START_FY=START_FY.ed()
        END_FY=absorb_date(dt_in=datetime.date(int(cr.FY),9,30))
        END_FY=END_FY.ed()

        BEGELG_DT=None
        #Determine when this person became eligible
        if cr.BEGELG_DT!="":
            #if there's an entry in this row
            BEGELG_DT=sasd2ed(int(cr.BEGELG_DT))
        elif len(self.years)>1:
            #If they were in a previous cohort file, assume they started at the
            #beginning of this year
            BEGELG=START_FY
        else:
            #if there's no record of their first elig and no previous mention
            #then assume they started at the beginning of this FY.  This is
            #imperfect, but should do well for bens who joined before FY03
            BEGELG=START_FY

        #not everyone has an end of eligibility specified
        ENDELG_DT=None
        if cr.ENDELG_DT!="":
            ENDELG_DT=sasd2ed(int(cr.ENDELG_DT))

        startdep=None
```

```
        if cr.startdep != "":
            startdep=absorb_date(sd_in=cr.startdep)
            startdep=startdep.ed()

        enddep=None
        if cr.enddep   != "":
            enddep=absorb_date(sd_in=cr.enddep)
            enddep=enddep.ed()

        BENCAT_DT=None
        if cr.BENCAT_DT!= "":
            BENCAT_DT=sasd2ed(int(cr.BENCAT_DT))


        AREA_DT=None
        if cr.AREA_DT  != "":
            AREA_DT=sasd2ed(int(cr.AREA_DT))

        AREA=int(cr.AREA)

        #Update years reflected
        self.years.append(cr.FY)

        #First, let's deal with any BENCAT information
        #update the bencat with the specified one from the row if need be
        if cr.BENCAT not in self.BENCAT or "UNKNOWN" in self.BENCAT:
            #here's our kludgy "Unnown BENCAT change" code.  More on this below
            if len(self.BENCAT)>0:
                if self.BENCAT[-1]=="UNKNOWN":
                    #print "making kludgy update... new bencat is", cr.BENCAT
                    #self.print_full_bio()
                    #We can now replace the unknown change over
                    del self.BENCAT[-1]
                    #Then we have to replace the unknown timeline entry with the
                    #now known one.
                    for entry in self.TimeLine:
                        if entry[1]=="UNKNOWN":
                            u_date=entry[0]
                            del self.TimeLine[self.TimeLine.index(entry)]
                            self.TimeLine.append((u_date,cr.BENCAT))
                            self.valid()
                            break
                    #print "kludge update made"
                    #self.print_full_bio()
            self.BENCAT.append(cr.BENCAT)

        #if there's no entry showing when they became this bencat, make one
        if cr.BENCAT not in [change[1] for change in self.TimeLine]:
            self.TimeLine.append((max(BEGELG_DT,START_FY),cr.BENCAT))

        #If this person changed beneficiary categories (for example, by retiring)
        #then we need to make a note of that.  This is indicated in the field
        #BENCAT_CHG.  1=No Change, 2=AD-->RET, 3=Dependent to DR, 4=GRD/RES mixed,
        #5=ADFM to survivor, 6=Other
        if cr.BENCAT_CHG==1: pass
        elif cr.BENCAT_CHG==2 and BENCAT_DT and cr.BENCAT in ["ACT","GRD"]:
```

```
            self.TimeLine.append((BENCAT_DT,"RET"))
            self.BENCAT.append("RET")
        elif cr.BENCAT_CHG==3 and BENCAT_DT and cr.BENCAT in ["DA","DGR"]:
            self.TimeLine.append((BENCAT_DT,"DR"))
            self.BENCAT.append("DR")
        elif cr.BENCAT_CHG!=1 and BENCAT_DT==None:
            #This person will change bencats next year, and we'll just deal with
            #it then.  There should be a BENCAT_DT in next years file.
            pass
        elif cr.BENCAT_CHG!=1 and BENCAT_DT!=None:
            #This person may change BENCAT in the next year to an unspecified
            #category.  This is a kludge, but what we'll do is add an entry for
            #and "UNKNOWN" BENCAT which will be modified the next year if
            #there is a next
            self.TimeLine.append((BENCAT_DT,"UNKNOWN"))
            self.BENCAT.append("UNKNOWN")
            #print "Unknown BENCAT_CHG from", cr.BENCAT, "during this year"
        else:
            print "No idea how to update this one"
            print cr
            self.print_full_bio()


    #Then let's deal with their whereabouts
        #let's also determine that person's first known whereabouts.  That'll be the
        #AREA they have listed.

        #let's see which areas they've been to
        ordered_AREAs=[entry[1] for entry in self.TimeLine if type(entry[1]) is int]

        #if the current AREA isn't in that list:
        if AREA not in ordered_AREAs:
            if AREA_DT:
                #take the later of the dates (date of move, date of first elig)
                self.TimeLine.append((max(BEGELG_DT,AREA_DT),AREA))
            else:
                #assume they got there at max (beginning of the FY, start of elig)
                self.TimeLine.append((max(BEGELG_DT,START_FY),AREA))
        #if it IS in the list but it's NOT the last place they've been
        elif AREA in ordered_AREAs and ordered_AREAs[-1]!=AREA:
            #same as above, but in a separate statement to prevent indexing
            #an empty list
            if AREA_DT: self.TimeLine.append((max(BEGELG_DT,AREA_DT),AREA))
            else: self.TimeLine.append((max(BEGELG_DT,START_FY),AREA))
        #if it's in the list, and it's the last entry, then we don't need to update
        else: pass

    #now, let's add any deployments that person may have experienced
        if startdep:
            self.TimeLine.append((startdep,"d"))
            self.TimeLine.append((enddep,"r"))

    #finally, let's make sure we place an eor in the history
        #if the person has an ENDELG date before the end of the FY, we'll put
        #that in the timeline.
        #
```

```
        #if more things happen after that date, this entry will be automatically
        #removed during validation.
        #
        #If they don't have an ENDELG, we'll put in an end of record for this FY
        #so that if they don't appear in the next FY, we know when we last saw them.
        if ENDELG_DT:
            self.TimeLine.append((min(END_FY,ENDELG_DT),"eor"))
        else:
            self.TimeLine.append((END_FY,"eor"))

        self.valid()

    def print_full_bio(self,indent=True):
        if indent: indention= "    "
        else: indention=""

        print indention+"ID:      "+self.id
        print indention+"famid:   "+self.famid
        print indention+"Service "+str(self.service)
        print indention+"BENCAT  "+str(self.BENCAT)
        print indention+"Timeline"
        for change in self.TimeLine:
            change_date=absorb_date(ed_in=change[0])
            print indention+indention+str(change[1])+" - on "+ change_date.sd() + " " +str(change_date.ed())
        print ""
        print ""


    def status_on_date(self,e_date):
        #returns this person's status on the absorb_date provided
        #if the date falls entirely outside of the person's record,
        #an error is returned.
        status_changes=[entry for entry in self.TimeLine if type(entry[1])==str or type(entry[1])==unicode]

        current_bencat="None yet"
        current_status="None yet"
        last_return_date=0 #0 if never returned from a deployment, date once returned
        deployments=0 #incremented at each return

        error_flag=False
        for entry in status_changes:
            if entry[0]>e_date: break #this update is after the date we're querying
            elif entry[0]==e_date and entry[1]=="eor": break

            if entry[1] in all_bencats:
                current_bencat=entry[1]
                if current_status =="None yet": current_status="pre"
                if type(current_status) is int: current_status="post"
            elif entry[1]=="d": current_status="during"
            elif entry[1]=="r":
                deployments+=1
                if current_bencat in dep_bencats:
                    last_return_date=entry[0]
                    current_status=1000*deployments+min(366,e_date-entry[0])
```

```
                else:
                    current_status="post"
            elif entry[1]=="eor":
                error_flag=True

        if "None yet" in [current_bencat, current_status]: error_flag=True
        if e_date>max([entry[0] for entry in status_changes]):
            #if the date in question is after the end of this record
            error_flag=True
        return (current_bencat, current_status, error_flag)

    def installation_on_date(self,e_date):
        installation_changes=[entry for entry in self.TimeLine if type(entry[1]) is int]

        current_location='Unknown Location'
        for entry in installation_changes:
            if entry[0]>e_date: break
            else: current_location=entry[1]

        return current_location

    def unclear(self):
        #a record is unclear if it begins with a deployment
        if self.TimeLine[0][1]=="d":
            return True
        #a record is unclear if there's a deployment when the person's not
        #in a deploying BENCAT
        last_known_bencat=0
        for entry in self.TimeLine:
            if entry[1] in all_bencats: last_known_bencat=entry[1]
            elif entry[1] == "d" and last_known_bencat not in dep_bencats: return True
            elif entry[1] == "UNKNOWN": return True
        if not self.valid(): return True

        #latest_year=None
        ##a record is unclear if it has discontiguous cohort entries
        #for year in self.years:
        #    if latest_year==None or latest_year==year-1: latest_year=year
        #    else: return True
        return False

    def personal_dps(self,start_date,last_date):
        #this returns the days this person spent in each status during
        #the specified interval
        #
        #formatted as p_dps[BENCAT][status]=days spent
        #so that changes in BENCAT can be reflected

        p_dps={}
        window=(start_date,last_date)
        #print window

        #Determine the nature of each interval in a person's history, then
        #add that interval to their p_dps (personal days per status)

        status_changes=[entry for entry in self.TimeLine if type(entry[1]) is str or type(
```

```
        entry[1]) is unicode]
    #print status_changes

    current_bencat=0
    ever_deployed=0 #zero or the last return date
    deployments=0 #incremented upon return
    currently_deployed=False #binary flag to see if the deployment ended yet

    #account for all time within the history's events
    for index in range(0,len(status_changes)-1):
        #let's see update the current bencat if we have to
        #print "Evaluating", status_changes[index], "to", status_changes[index+1]
        if status_changes[index][1] in all_bencats:
            #update this, add it to the personal dictionary
            current_bencat=status_changes[index][1]
            if current_bencat not in p_dps: p_dps[current_bencat]={}
            #if this person has never deployed, then let's count "pre" status
            if not ever_deployed:
                p_dps[current_bencat]["pre"]=p_dps[current_bencat].get("pre",0)+overlap(
                    window,(status_changes[index][0],status_changes[index+1][0]))
            elif currently_deployed:
                p_dps[current_bencat]["during"]=p_dps[current_bencat].get("during",0)+
                    overlap(window,(status_changes[index][0],status_changes[index+1][0])
                    )
            else:
                p_dps[current_bencat]["post"]=p_dps[current_bencat].get("post",0)+
                    overlap(window,(status_changes[index][0],status_changes[index+1][0])
                    )
        #we're looking at intervals, not entries, so only look at n-1 status_changes
        elif status_changes[index][1]=="d" and status_changes[index+1][1] in ["eor","r"
            ]:
            #this interval reflects a deployment
            ever_deployed=1
            currently_deployed=True
            p_dps[current_bencat]["during"]=p_dps[current_bencat].get("during",0)+
                overlap(window,(status_changes[index][0],status_changes[index+1][0]))
        elif status_changes[index][1]=="d" and status_changes[index+1][1] in all_bencats
            :
            #the next thing recorded after this person's deployment is a change in
                BENCAT
            #so, we'll count the "during" portion for this bencat for this window
            ever_deployed=1
            currently_deployed=True
            p_dps[current_bencat]["during"]=p_dps[current_bencat].get("during",0)+
                overlap(window,(status_changes[index][0],status_changes[index+1][0]))
        elif status_changes[index][1]=="r":
            currently_deployed=False
            deployments+=1
            #print "  has now returned from", deployments, "deployments"
            if current_bencat in dep_bencats:
                #then increment the statuses until this person has returned, or their
                    record ends
                for delta in [x for x in range(0,status_changes[index+1][0]-
                    status_changes[index][0]+1)]:
                    status_bin=1000*deployments+min(delta,366)
                    #print "  adding day to delta",deployments, delta,"in status bin",
```

```
                            status_bin
                        p_dps[current_bencat][status_bin]=p_dps[current_bencat].get(status_bin,0)+overlap(window,(status_changes[index][0]+delta,status_changes[index][0]+delta))
                else:
                    #this person returned after changing category, so we'll book it as a post for the new BENCAT.
                    p_dps[current_bencat]["post"]=p_dps[current_bencat].get("post",0)+overlap(window,(status_changes[index][0],status_changes[index+1][0]))
            else:
                print "Don't know what to do with the window following", status_changes[index][0], "for"
                self.print_full_bio()

        #for cleanliness, don't return k:v pairs with value=0
        for bencat in p_dps.keys():
            for status in p_dps[bencat].keys():
                if p_dps[bencat][status]==0:
                    del p_dps[bencat][status]
        return p_dps

    def arrival(self):
        first=min([entry[0] for entry in self.TimeLine])
        if first==None:
            print "Problem: There doesn't appear to be anything in this person's timeline. Or, \'None\' is a date in our TimeLine"
            print self.id
            print self.TimeLine
            self.print_full_bio()
        return first

    def departure(self):
        last= max([entry[0] for entry in self.TimeLine])
        return last
```

B.2.13 simple_demand_maker.py

```
#This allows the user to create a dummy file that can seed arena with trivial
#demand patterns.
import csv
import os
from optparse import OptionParser
from numpy.random import poisson

def make_simple_seed(location):
    fhandle=open(location+"simple_arena_seed.csv","wb")
    o=csv.writer(fhandle)

    #Specify a set of installations.
    installations=(1662,24,110,48,49,60,64,109)
    num_inst=len(installations)

    #Specify the number of patients arriving each day in arena.
    daily_arrivals=1000

    #A list of tuples is used to define a step function.  The first value of each ?
    #tuple is the day on which that demand pattern starts.  The second value is
    #another tuple that shows the patient demand for each site.  Each site is
    #represented by a tuple (ACT demand, DA demand)
    ACT_base=36
    DA_base=20

    pattern=[]

    deterministic=False

    pattern.append((0000,(\
                             (9.14,1.69),\
                             (8.15,1.09),\
                             (35.61,20.38),\
                             (12.64,6.45),\
                             (11.13,5.74),\
                             (20.31,13.23),\
                             (1,1),\
                             (1,1)\
                               )))

    pattern.append((360,()))


    #Now, iterate through this pattern and output the lines to a csv.file
    sn=0
    for interval in range(0,len(pattern)-1):
        for day in range(pattern[interval][0],pattern[1+interval][0]):
            lines_written=0
            for site in range(num_inst):
                ACT_arrivals=pattern[interval][1][site][0]
                if not deterministic: ACT_arrivals=poisson(ACT_arrivals)
                for ACTarrival in range(int(ACT_arrivals)):
                    o.writerow([day,day,sn,installations[site],"ACT"])
                    sn+=1
```

```
                    lines_written+=1
                DA_arrivals=pattern[interval][1][site][1]
                if not deterministic: DA_arrivals=poisson(DA_arrivals)
                for DAarrival in range(int(DA_arrivals)):
                    o.writerow([day,day,sn,installations[site],"DA"])
                    sn+=1
                    lines_written+=1
            while lines_written<daily_arrivals:
                o.writerow([day,day,sn,"0","O"])
                sn+=1
                lines_written+=1

if __name__ == '__main__':
    parser=OptionParser()
    parser.add_option("-v","--verbose",action="store_true",dest="verbose",default=False)
    parser.add_option("-t","--temp",action="store_true",dest="use_temp_dir",default=False)
    parser.add_option("-s","--subfolder",action="store",type='string',dest="sample_dir",
        default="")
    (options,args)=parser.parse_args()

    sample_dir=options.sample_dir
    verbose=options.verbose
    use_temp_dir=options.use_temp_dir

    location=""
    if use_temp_dir:
        location = ('/Users/johnhess/Desktop/DataCubeTemp/' + sample_dir+"/")
    else:
        location = ('/Volumes/Data/DataCube/nas/data/batch_csv/' + sample_dir+"/")

    make_simple_seed(location)
```

B.2.14 timeline_parser.py

```
"""
This file contains a few functions.

The first, days_per_status, returns the
total number of days each person in the set of histories passed to it spent in
each status.

print_all_histories is a simple function for printing everything passed to it
(useful for debugging)

overlap is a simple function for calculating the overlap of two time windows

personal_dps is used to determine the number of days a person spent in each
status (e.g. 100 days "pre" deployment, 300 "during" deployment, 1 in the first
day after deployment, 1 in the second day after deployment, etc.).

This function is used instead of the history class' member function "status on
date" because calling that for each day takes a very long time. This should
almost certainly have been made a member function of the history class itself.
"""



from datehandler import absorb_date
import datetime
from jth_tools import Printer
import cPickle
import os
import time

all_bencats=["ACT","RET","GRD","IGR","DA","DR","DS","DGR","IDG","OTH","Z"]
dep_bencats=["ACT","GRD"]

def days_per_status(h,sd,ld,l=None,f=None,save=False):
    #pull master_dps from file if we've got one
    if "dps.cpickle" in os.listdir(l+f):
        print "...found dps.cpickle"
        print "...loading that instead of recomputing from histories"
        master_dps=cPickle.load(open(l+f+"dps.cpickle",'rb'))
        return master_dps

    master_dps={}
    num_persons=0
    lasttime=0
    for person in h:
        num_persons+=1
        if int(time.time())!=lasttime:
            Printer(str(num_persons) + " Individual histories added to the 'days per status'
                counters")
            lasttime=int(time.time())
        p_dps=personal_dps(h[person],sd,ld)
        for bencat in p_dps:
            if bencat not in master_dps: master_dps[bencat]=p_dps[bencat]
            else:
```

```
                for status in p_dps[bencat]:
                    master_dps[bencat][status]=master_dps[bencat].get(status,0)+p_dps[bencat][status]
    Printer(str(num_persons) + " Individual histories added to the 'days per status' counters")
    print "" #to get a newline after the Printer use

    if save:
        print "Making intermediate data file dps.cpickle"
        o=open(l+f+"dps.cpickle",'wb')
        dumper=cPickle.Pickler(o)
        dumper.fast=True
        dumper.dump(master_dps)
        print "Finished dumping file"

    return master_dps

def print_all_histories(h):
    for person in h:
        print person
        h[person].print_full_bio(indent=True)

def overlap(a,b):
    return max(0,min(a[1],b[1])-max(a[0],b[0])+1)

def personal_dps(person,start_date,last_date):
    #this returns the days this person spent in each status during
    #the specified interval
    #
    #formatted as p_dps[BENCAT][status]=days spent
    #so that changes in BENCAT can be reflected

    p_dps={}
    start_date=absorb_date(dt_in=start_date)
    last_date=absorb_date(dt_in=last_date)
    window=(start_date.ed(),last_date.ed())

    #Determine the nature of each interval in a person's history, then
    #add that interval to their p_dps (personal days per status)

    status_changes=[entry for entry in person.TimeLine if type(entry[1])==str]

    current_bencat=0
    ever_deployed=0 #zero or the last return date
    currently_deployed=False #binary flag to see if the deployment ended yet

    #account for all time within the history's events
    for index in range(0,len(status_changes)-1):
        #let's see update the current bencat if we have to
        if status_changes[index][1] in all_bencats:
            #update this, add it to the personal dictionary
            current_bencat=status_changes[index][1]
            if current_bencat not in p_dps: p_dps[current_bencat]={}
            #if this person has never deployed, then let's count "pre" status
            if not ever_deployed:
                p_dps[current_bencat]["pre"]=p_dps[current_bencat].get("pre",0)+overlap(
```

```
                    window,(status_changes[index][0],status_changes[index+1][0]))
            elif currently_deployed:
                p_dps[current_bencat]["during"]=p_dps[current_bencat].get("during",0)+
                    overlap(window,(status_changes[index][0],status_changes[index+1][0]))
            else:
                p_dps[current_bencat]["post"]=p_dps[current_bencat].get("post",0)+overlap(
                    window,(status_changes[index][0],status_changes[index+1][0]))
        #we're looking at intervals, not entries, so only look at n-1 status_changes
        elif status_changes[index][1]=="d" and status_changes[index+1][1] in ["eor","r"]:
            #this interval reflects a deployment
            ever_deployed=1
            currently_deployed=True
            p_dps[current_bencat]["during"]=p_dps[current_bencat].get("during",0)+overlap(
                window,(status_changes[index][0],status_changes[index+1][0]))
        elif status_changes[index][1]=="d" and status_changes[index+1][1] in all_bencats:
            #the next thing recorded after this person's deployment is a change in BENCAT
            #so, we'll count the "during" portion for this bencat for this window
            ever_deployed=1
            currently_deployed=True
            p_dps[current_bencat]["during"]=p_dps[current_bencat].get("during",0)+overlap(
                window,(status_changes[index][0],status_changes[index+1][0]))
        elif status_changes[index][1]=="r":
            currently_deployed=False
            if current_bencat in dep_bencats:
                #then increment the statuses until this person has returned, or their record
                     ends
                for delta in range(0,status_changes[index+1][0]-status_changes[index][0]+1):
                    p_dps[current_bencat][delta]=p_dps[current_bencat].get(delta,0)+overlap(
                        window,(status_changes[index][0]+delta,status_changes[index][0]+
                        delta))
            else:
                #this person returned after changing category, so we'll book it as a post
                    for the new BENCAT.
                p_dps[current_bencat]["post"]=p_dps[current_bencat].get("post",0)+overlap(
                    window,(status_changes[index][0],status_changes[index+1][0]))
        else:
            print "Don't know what to do with the window following", status_changes[index
                ][0], "for",person
            person.print_full_bio()

    #for cleanliness, don't return k:v pairs with value=0
    for bencat in p_dps.keys():
        for status in p_dps[bencat].keys():
            if p_dps[bencat][status]==0:
                del p_dps[bencat][status]
    return p_dps
```

C Site to Site Variation in Episode Length

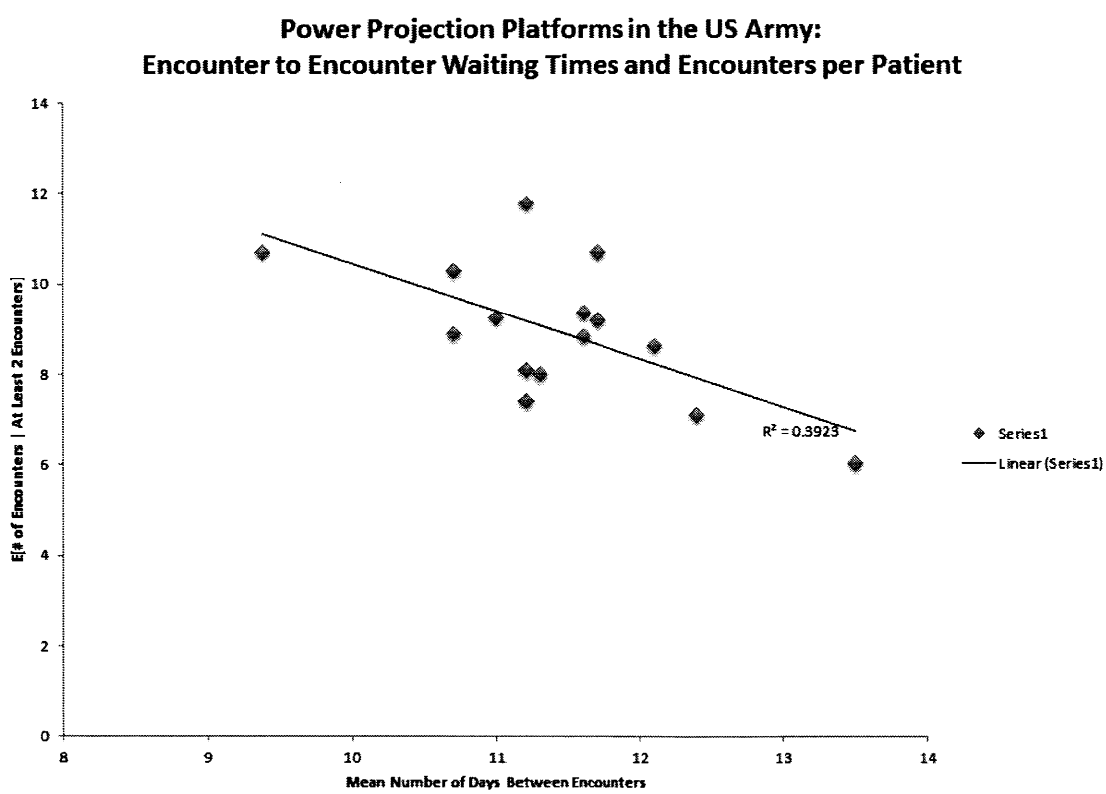
Across different sites – even across similar sites like Army Power Projection Platforms (PPPs) – the average number of encounters per episode varies greatly.

When a system is overburdened, we expect increased caseloads. With larger caseloads, providers with a limited amount of time may meet more infrequently with their patients or provide fewer encounters to each patient. A cursory examination shows significant variation in episode length and in care frequency. Such variation is not readily explained.

Looking at Army Power Projection Platforms, installations which see patients less frequently also provide less encounters per episode (see Figure 51 on page 188).

This does not mean that such sites are necessarily offering a lower standard of care. But a correlation between these two variables does warrant further investigation because it suggests that they may both be affected by caseload size. We did not have the data to examine this third variable.

Figure 51: Slight Correlation between Quantity of Care per Patient and Time Between Encounters, Army Power Projection Platforms



References

[MHS, 2008] (2008). Military Health System's Guide to Access Success.

[GAO, 2010] (2010). Military personnel: Enhanced collaboration and process improvements needed for determining military treatment facility personnel requirements. Technical report, Government Accountability Office.

[MHS, 2011] (2011). Department of defense task force on military health system governance final report.

[NIH, None] (None). Post traumatic stress disorder. http://www.ncbi.nlm.nih.gov/pubmedhealth/PMH0001923/.

[Aday and Andersen, 1974] Aday, L. A. and Andersen, R. (1974). A framework for the study of access to medical care. *Health Services Research*, 9(3):208–220.

[Army, 2012] Army (2012). Tele-behavioral health operations manual. Technical report, U.S. Army Medical Command.

[Arthur et al., 2007] Arthur, D. C., MacDermid, S., and Kiley, K. C. (2007). An achievable vision: Report of the department of defense task force on mental health. Technical report, Defense Health Board, Falls Church, VA.

[Associated Press, 2008] Associated Press (2008). Remote-control warriors suffer war stress. http://www.usatoday.com/news/topstories/2008-08-07-3990864370_x.htm.

[ATA, None] ATA (None). American telemedicine association: What is telemedicine?

[Atkinson et al., 2009] Atkinson, M. P., Guetz, A., and Wein, L. M. (2009). A dynamic model for ptsd among u.s. troops in oif.

[Belasco, 2009] Belasco, A. (2009). Troop levels in the afghan and iraq wars, fy2001-fy2012: Cost and other potential issues. Technical report, Congressional Research Service.

[Berg and Denton, 2012] Berg, B. and Denton, B. (2012). *Handbook of Healthcare System Scheduling, Ch. 6.* Number 168 in International Series in Operations Research and Management Science. Springer Science and Business Media.

[Brusher, None] Brusher, E. A. (None). Combat and operational stress control. http://www.bordeninstitute.army.mil/published_volumes/combat_operational/CBM-ch4-final.pdf.

[Carabajal, 2012] Carabajal, S. (2012). Army expanding successful embedded behavioral health program. www.army.mil/article/69479.

[Card et al., 2008] Card, D., Dobkin, C., and Maestas, N. (2008). The impact of nearly universal insurance coverage on health care utilization: Evidence from medicare. *American Economic Review*, 98(5):2242–2258.

[Casscells, 2007] Casscells, S. W. (2007). Mhs human capital strategic plan. Technical report, Military Health System.

[Chappelle et al., 2010] Chappelle, W., McDonald, K., and King, R. E. (2010). Psychological attributes critical to the performance of mq-1 predator and mq-9 reaper u.s. air force sensor operators. Technical report, 711th Human Performance Wing, Air Force Research Laboratory.

[Cooke, 2003] Cooke, J. A. (2003). Manpower requirements determination in the institutional army. Strategy research project, USAWC.

[Dial et al., 1995] Dial, T. H., Palsbo, S. E., Bergsten, C., Gabel, J. R., and Weiner, J. (1995). Clinical staffing in staff- and group-model hmos. *Health Affairs*, 14(2):168–180.

[Faulkner and Goldman, 1997] Faulkner, L. R. and Goldman, C. R. (1997). Estimating psychiatric manpower requirements based on patients' needs. *Psychiatric Services*, 48(5):666–670.

[Garb, None] Garb, H. (None). Background paper on lackland behavioral health questionnaire results (working paper). Technical report.

[Greenberg and Cultice, 1997] Greenberg, L. and Cultice, J. M. (1997). Forecasting the need for physicians in the united states: The health resources and services adminstration's physician requirements model. *Health Services Research*, 31(6):723–737.

[Hablen, None] Hablen, J. (None). What is PTSD? A Handout from the National Center for PTSD.

[Hailey et al., 2008] Hailey, D., Roine, R., and Ohinmaa, A. (2008). The effectiveness of telemental health applications: A review. *Canadian Journal of Psychiatry*, 53(11):769–778.

[Harris et al., 2010] Harris, D. M., Marr, L. M., Borsky, A. E., and Bickett, T. A. (2010). The psychological health risk-adjusted model for staffing (phrams): Development, operation, projections and user application. Technical report, CNA.

[Hess et al., 2011] Hess, J., Kamin, C., and Kenley, C. R. (2011). Psychological health in the united states military: Making sense of what we know. INCOSE.

[Hess and Srinivasan, 2012] Hess, J. and Srinivasan, J. (2012). Case study: Behavioral health delivery system site b (working). Technical report, PTSI, LAI, MIT.

[Hoge et al., 2004] Hoge, C. W., Castro, C. A., Messer, S. C., McGurk, D., Cotting, D. I., and Koffman, R. L. (2004). Combat duty in afghanistan, mental health problems, and barriers to care. *The New England Journal of Medicine*, 351(1):13–22.

[J-MHAT, 2011] J-MHAT (2011). Joint mental health advisory team 7 (j-mhat 7) operation enduring freedom 2010 afghanistan. Technical report, Office of the Surgeon General United States Army Medical Command, Office of the Command Surgeon HQ, USCENTCOM and Office of the Command Surgeon US Forces Afghanistan (USFOR-A).

[Johnson and Sherman, 2007] Johnson, S. J. and Sherman, M. D. (2007). The psychological needs of u.s. military service members and their families: A preliminary report. Technical report, Military Deployment Services Task Force.

[Keeler et al., 1986] Keeler, E. B., Wells, K. B., Manning, W. G., Rumpel, J. D., and Hanley, J. M. (1986). The demand for episodes of mental health services. Technical Report ISBN: 0-8330-0770-X, RAND, Santa Monica, CA.

[Levy et al., 2009] Levy, R. A., Borsky, A., and Hill, C. (2009). The tricare mental health/substance abuse benefit and its comparability to private and public plans. Technical Report CRM D0021255.A2, CNA.

[Levy and Gabay, 2008] Levy, R. A. and Gabay, M. (2008). Does tricare have enough physicians and other providers for the non-enrolled population. Technical report, CNA.

[Mansfield, 2010] Mansfield, A. J. (2010). Deployment and the use of mental health services among u.s. army wives. *New England Journal of Medicine*, 362(2):101–109.

[Meredith, 2011] Meredith, L. S. (2011). Promoting psychological resilience in the u.s. military. Technical report, RAND Center for Military Health Policy Research.

[MHS, 2010] MHS (2010). Military health system overview of strategic performance measures (presentation).

[Milliken et al., 2007] Milliken, C. S., Aucherlonie, J. L., and Hoge, C. W. (2007). Longitudinal assessment of mental health problems among active and reserve component soldiers returning from the iraq war. *Journal of the American Medical Association*, 298(18):2141–2148.

[National Institutes of Mental Health,] National Institutes of Mental Health. Post Traumatic Stress Disorder.

[Niebuhr et al., 2010] Niebuhr, D. W., Gubata, M. E., and Yuanzhang, L. (2010). Accession and medical standards analysis and research activity 2010 annual report. Technical report, AMSARA, Silver Springs, MD.

[Piver-Renna, 2010] Piver-Renna, J. M. (2010). Program consultation (procon) part i: Retrospective evaluation of a mobile behavioral health service in garrison. Technical report, U.S. Army Public Health Command.

[Rosenheck and Fontana, 2007] Rosenheck, R. A. and Fontana, A. F. (2007). Recent trends in va treatment of post-traumatic stress disorder and other mental disorders. *Health Affairs*, 26(6):1720–1727.

[Sackett and Mavor, 2006] Sackett, P. R. and Mavor, A. S. (2006). Assessing fitness for military enlistment: Physical, medical, and mental health standards. Technical report, National Research Council.

[Scott et al., 2011] Scott, S., Ippolito, A., and Srinivasan, J. (2011). Case study: Behavioral health delivery system site a. Technical report, PTSI, LAI, MIT.

[Scott and Srinivasan, 2012] Scott, S. and Srinivasan, J. (2012). Case study: Behavioral health delivery system site e. Technical report, PTSI, LAI, MIT.

[Seal et al., 2009] Seal, K. H., Metzler, T. J., Gima, K. S., Berthenthal, D., Maguen, S., and Marmar, C. R. (2009). Trends and risk factors for mental health diagnoses among iraq and afghanistan veterans using department of veterans affairs health care, 2002-2008. *American Journal of Public Health*, 99(9):1651–1658.

[Spiegel, 2008] Spiegel, P. (2008). Battle breaks out over the predator. *Los Angeles Times.*

[Stewart, 2011] Stewart, P. (2011). Overstretched u.s. drone pilots face stress risk.

[Tanielian and Jaycox, 2008] Tanielian, T. and Jaycox, L. H. (2008). Invisible wounds of war. Technical report, RAND.

[Thomas et al., 2010] Thomas, J. L., Wilk, J. E., Riviere, L. A., McGurk, D., Castro, C. A., and Hoge, C. W. (2010). Prevalence of mental health problems and functional impairment among active component and national guard soldiers 3 and 12 months following combat in iraq. *Archives of General Psychiatry*, 67(6):614–623.

[Thomsen, 1999] Thomsen, K. N. (1999). Health promotion and wellness staffing methods. Master's thesis, U.S. Army-Baylor University Graduate Program in Health Care Administration.

[Utley and Worthington, 2012] Utley, M. and Worthington, D. (2012). *Handbook of Healthcare System Scheduling, Ch. 2.* Number 168 in International Series in Operations Research and Management Science. Springer Science and Business Media.

[Weinick, 2011] Weinick, R. M. (2011). Programs addressing psychological health and care for traumatic brain injury among u.s. service members and their families. Technical report, RAND National Defense Research Institute.

[Wells, 2001] Wells, K. (2001). Ethnic disparities in unmet need for alcoholism, drug abuse, and mental health care. *American Journal of Psychiatry*, 158(12):2027–2032.

[Wennberg, 2004] Wennberg, J. E. (2004). Practice variations and health care reform: Connecting the dots. *Health Affairs*, Web Exclusive.

[Wennberg et al., 2002] Wennberg, J. E., Fisher, E. S., and Skinner, J. S. (2002). Geography and the debate over medicare reform. *Health Affairs*, Web Exclusive.

[Woodson, 2011] Woodson, J. (2011). Memorandum for assistant secretaries of the army, navy, and air force: Tricare policy for access to care.