Cognitive monitoring and motor speed assessment using unobtrusive computer keyboard measures

By

Edward Anthony VanBaak

Master's Capstone Project

Presented to the Department of Medical Informatics and Clinical Epidemiology
and the Oregon Health & Science University School of Medicine
in partial fulfillment of the requirements for the degree of Master of Biomedical Informatics

June 2009

School of Medicine

Oregon Health & Science University

Certificate of Approval

This is to certify that the Master's Capstone Project of

Edward A. VanBaak

"Cognitive monitoring and motor speed assessment using unobtrusive computer keyboard measures."

Has been approved

____

Capstone Advisor – Holly Jimison, Ph.D.

Table of Contents

Acknowledgements

I would like to express my sincere gratitude and thanks to my advisor Dr. Holly Jimison without whose patience and guidance I would not have been able to complete this project. Many thanks to Dr. Misha Pavel for his encouragement as we aim to improve the lives of other through our work.

Thanks to the members of my research group who challenged my ideas throughout my time at OHSU: Kyle Ambert, William Hatt, Chia Hua Yu, and Zephy McKanna. Thanks to OHSU and the Department of Medical Informatics for many wonderful friendships and experiences.

I would also I like to recognize and thank all of the elderly participants in our study. Without their cooperation and understanding none of this would have been possible.

This work was supported by the following National Institute on Aging (NIA) Grants; Oregon Roybal Center for Aging and Technology P30AG024978 – (for use of ORCATECH pilot grants, living laboratory, and/or administrative core support); Oregon Alzheimer's Disease Center (OADC) P30AG08017 – (for use of BRP/ORCATECH clinical data); Bioengineering Research Partnership, Oregon Roybal Center for Aging and Technology R01AG024059 – (for use of BRP data).

Abstract

Many elderly people are affected by a decline in cognitive abilities as they age. While commonly linked to general aging processes, this decline can be indicative of disease [1]. Cognitive decline can occur as a result of natural causes but may become exacerbated by other physiological diseases such as Parkinson's disease [2]. The ability to monitor and detect cognitive decline has diagnostic and clinical benefits [3]. This research looks to study and monitor cognitive decline through motor speed changes, as they may predict cognitive and also functional decline [4-8]. I hypothesize that keystroke speed as determined by a standard unobtrusive computer keyboard typing test can be used as a remote surrogate for a Halstead-Reitan finger-tapping test commonly used in neuropsychological testing, and that the resulting motor skill assessment will strongly correlate with cognitive function among the elderly as observed in initial assessment of our multiyear longitudinal study. Our results indicate a correlation between our typing speed assessment and results of the finger tapping test.

Introduction

The decline of cognitive abilities can be indicative of general aging or of disease. [1] It is important to be able to differentiate aging processes from disease processes. The onset of disease can be slow and progress over many years. Aging, almost by definition, is an extended continual process. While there are disease processes which affect cognitive abilities acutely, this research was done in order to further the science dealing with long term chronic processes affecting cognitive function.

Cognitive decline is a broad term for a variety of degenerative processes affecting the mental and neurological capacities of individuals. These processes can be rather benign and limited to normal aging processes. Cognitive decline can also include mild cognitive impairment (MCI) which lies on the degenerative spectrum in between normal aging and a diagnosis of dementia [9-11]. There has been a large variety of other terms for describing this phase in between normalcy and confirmed dementia [11-17]. Cognitive decline is a broad term which covers the spectrum from general to debilitating diseases like Alzheimer's disease (AD). Persons with MCI have a 10% to 15% risk per year of developing dementia as compared with the risk of the general population of 1% - 2% per year [18]. Dementia can dramatically affect many cognitive functions including memory, problem solving, and language skills. Disorientation is another symptom which can place a large burden on caretakers and those suffering from dementia.

This research attempts to determine if one unobtrusively monitored activity, typing on a computer keyboard, correlates with the common neurological test of finger tapping. Ideally, a person’s condition could be monitored for changes in cognitive decline and the cause of that decline attributed to either normal aging or disease.

The finger tapping test (FTT) is designed as a simple method to gauge motor speed. [19] Motor function is determined by a number of areas in the brain, however the motor strip rostral to the central sulcus in most important. [19] The FTT is designed to measure motor function with minimal influence from other cognitive processes. [20] The test involves a device with a lever mounted to a mechanical counter. One company, Reitan Neuropsychology Laboratory Inc., sells such a device as display in Figure 1. [21]

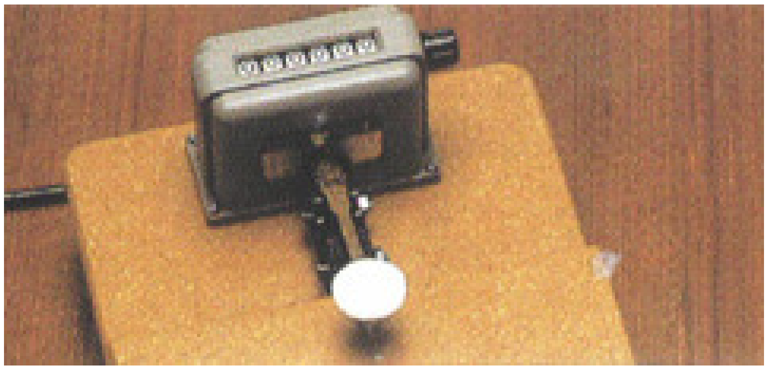


Figure 1. Mechanical Finger Tapping Counter.

This study analyzed finger tapping data from a standard finger tapping neuropsychological battery. A plethora of keyboard typing data was also collect, analyzed, and studied as part of this research. Methods of data gathering, cleaning, analysis and interpretation resulted in research discussed herein. The main goals of this project remained constant throughout this process, however the approach changed as the project progressed. This research did not start out with knowledge of how to answer the main research questions. The structure of the project allowed for creativity, ingenuity, hard work, and collaboration to help compose solutions to achieve progress.

Numerous papers have been previously published documenting work in the areas of this research. This study delves into the fields of Informatics, Neurology, Psychology, Aging, Medicine, and Technology. While the fields of Neurology and Psychology have been extant for a long time, they are certainly still active and making great discoveries today. The integration of technology in medicine has more recent roots, and is still working its way into modern medicine. A study such as this which explores unique aspects from a variety of fields is a massive undertaking which involves researchers from a wide array of backgrounds.

Several papers have been published recently which closely relate to the topic of this research. One recent article of note was published by Jimison and Pavel, et al, in 2004, *Unobtrusive monitoring of computer interactions to detect cognitive status in elders*. [23] Numerous other papers published by Pavel (2007), Jimison (2006), Hayes (2003), Jobbagy (2005) and Scarmeas (2005) have laid the recent foundation for this research. [22 - 27]

My work on this project began as an education experience. This work draws on many fields and areas of personal interest. There are certainly aspects of computer programming, information technology, medical informatics, healthcare, and home monitoring all combined with an academic research focus.

The overall goal of this study was to lay the ground work to determine whether the unobtrusive monitoring of general activity in the home can be used as a surrogate to detect changes in motor and cognitive function. The specific aim for this research was to determine if motor measures of keyboard interactions on the computer could serve as estimates of the standard neuropsychological test for finger tapping (FTT). It is hoped that early detection of MCI will lead to treatment earlier in the progression of disease. This would then lead to an increased quality of life through treatment of the symptoms and progression of disease.

Background

It is essential to monitor or observe people who may have cognitive decline in order to effectively diagnose and treat them. The ability to unobtrusively monitor patients in their home holds significant advantages as compared to a monitoring session at a study facility which is unfamiliar to the patient. This is especially important when dealing with cognitive decline. While there are some drawbacks, such as lack of environmental control, this research embraced the opportunity to use unobtrusively monitor subjects in their normal, comfortable home environment.

With the growing elderly population in the United States and around the world [28] there will be an increased need for preventative care for the aging population. This need stems both from our duty to provide them with increased quality of life [29], and the financial realities that we will face when caring for a rapidly growing population. [30,31] Previous studies [32 - 34] have shown the importance of early recognition of cognitive decline, but current tests are expensive, time consuming, and are administered infrequently.

Many neuropsychological tests have been developed in an attempt to diagnose and quantify MCI and dementia. Both aging and disease processes can lead to MCI or dementia. It is possible to miss the onset of disease and incorrectly attribute a decline in cognitive abilities to normal aging processes. The need for an elderly individual's condition to engage the medical community in search of a diagnosis can be disregarded. This need can go unnoticed by friends, caretakers, or family members if they spend only a limited amount of time with the elderly person. The elderly may not have sufficient cognitive power to realize the extent of the problem. This research hopes to advance the fields of science and medicine to improve the lives of others.

The benefit of the approach used in the BRP study, compared to traditional methods, is that the monitoring is continuous. The cost of the testing also drops as specially trained individuals are not required at the home in order to administer tests or monitor the situation. Of the more than 230 participants in the study 196 of them have Internet connected computers that are monitored as a part of the study. While the scale of the project promises to offer large quantities of useful information that can be used for later research it also poses a challenge to our ability to carefully manage and process the computer usage information. Additionally, we needed to control for several types of computer events that could lead to a misinterpretation of valid computer activity. This was due to the necessary but often unpredictable nature of the interactions taking place between the systems involved, the users, the operating system, and the other applications running on the computer. Comprehension of the foundation, or metadata, of the data must occur for information to be derived from analysis. This includes a grasp of the study design, database design, database metadata, and any changes made since implementation.

The Biomedical Research Partnership (BRP) is a NIA funded longitudinal research study involving over 230 computer-literate elderly participants. The study focuses on continuous and unobtrusive in-home assessment of physical activity and computer usage. [26] One of the aims of the study is to lay the ground work to determine whether the unobtrusive monitoring of general activity in the home can be used as a surrogate to detect changes in motor and cognitive function. Early detection of disease process could then lead to preventative treatment of disease to increase the patients quality of life. In order to make progress toward these goals, the subjects in our study must be observed to deliver longitudinal data thereby enabling the potential for accurate research. Currently physical activity is being monitored through motion sensors in the patient’s homes, and computer usage is being monitored through a program that is installed on participant’s personal computers.

The Computer Monitoring Software (CMS) that was used in the study was designed at OHSU's Division of Biomedical Engineering (BME). The software functions in two ways. Before the user logs onto the computer, the CMS prompts to enter a username and password set at the beginning of the study. During and after the entry of a username / password combination, the CMS monitors all computer activity.

Five types of events were recorded from the participants' computers (Trigrams, Logins, Login Passwords, Application Changes, and Mouse Events). Because of storage constraints each type of data has to be stored separately and then later pooled in a single file structure so that each type of event can be viewed in relation to the others. Initially, all the events were viewed to assess general activity and computer usage. Data was displayed to verify that during a computer usage session, and 5 types of events were recorded to the database.

Several graphing techniques were adapted for use in order to explore the data and look for errors. The host computers were uniformly equipped with Microsoft Windows XP, though hardware and peripherals such as mouse input devices were not as uniform. A primary concern of this investigation was to discover and resolve unexpected behavior that may occur between the operating system and the peripheral hardware. This was done to ensure that the information being gathered from the study participants could later be correctly interpreted. The login event is stored, as well as all of the keystrokes that were used to generate the password, and recorded with millisecond precision.

Once the user has logged in, the program also monitored various types of mouse input, such as movement and mouse click events. We also monitored application activity and recorded the path and time at which an application window gained focus on the user's screen. Although keyboard typing activity is recorded, this information is restricted and obfuscated over concerns for the patient's privacy.

Rather than recording actual keystrokes in the order they were entered, the keyboard data is recorded in the form of trigrams. Trigrams consist of the last 3 key codes that the user has entered, and the amount of time that has elapsed between the 3 key press events. The trigrams are also time stamped with a limited 1 hour resolution, which allows later researcher to have information about the approximate time the key information and how fast it was entered without the ability to recreate what was actually typed. Furthermore, key activity is only recorded when the participant is using a web browser or typing an e-mail in Microsoft Outlook.

Once a day, the information that was collected by the CMS was forwarded over a broadband Internet connection to a secure MySQL database server that is run by OHSU's Division of Biomedical Engineering (BME). The data is then securely stored to be used for current and future research. As much of the data will be used for longitudinal investigations, the amount collected throughout the course of the study will require a significant investment in data storage capability. The design of the back end data storage system is subject to a complex set of requirements.

It was important to learn about the structure and design of the data collected in this study. This research was built on the ORCATECH foundation. Their previous work provided, support staff, organizational structure, data, and provided a platform for this research project. The metadata and structure of the data, and how the data was collected was important to understand.

Each data type is stored in its own table, with the exception of the mouse data, which because of its size, is stored in a separate MySQL server at BME. Although this allowed us to work with very large data sets, and it scales well as the number of study participants increases, it also presents computational challenges. This current investigation, and undoubtedly later analyses, will require that all of the information for a specific user be collated into a format for easy and efficient analysis. Figure 2 displays the database schema for the data involved with this study.

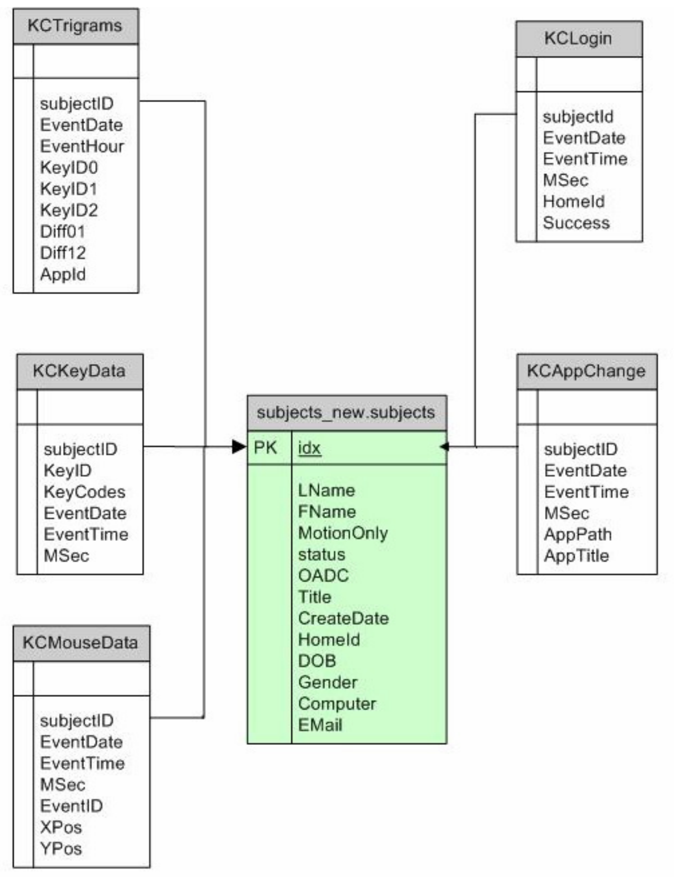


Figure 2. Database scheme of subject's computer usage data

Data is collected from an array of data monitoring equipment in the larger BRP study. The living areas of the BRP subjects have a plethora of sensors collecting data from movement and activity. Among the activities measured are appliance use; movement between rooms, movement as measured by weight sensors on the floor, motion sensors in the bed and also computer use activity.

This research focuses on the computer usage data. Of this data, there are 3 categories of activity, mouse, keyboard, and application usage. The application use was briefly observed, however it was not the focus of the research. The mouse data was briefly observed to determine if the keyboard and mouse were both used at the time of login. A cursory view of the data seemed to indicate that both the keyboard and mouse were used during the login window of time.

There are two collections of keyboard data. The larger is a database table (KCTrigrams) of each keystroke typed when the user was in one of these programs: Microsoft Word, Outlook Express, or Internet Explorer. These keystrokes were recorded in the database in such a way to map which key was pressed but in such a fashion to obfuscate the data and thus retain the subject's privacy and confidentiality.

The second, smaller keyboard database table (KCKeyData) set contains all the keystrokes pressed during a login session. The users were required to login into the computer before each session. This database is significantly smaller than the aforementioned database. This was the primary data set analyzed in this research project. Every keystroke pressed during the login session is recorded into the KCKeyData table of the BRP database. Any keystroke pressed during this session, or from the time the "Log On to Windows" window comes into focus, will be recorded. These included any key on the keyboard. While most of the keys were either [a-z], [A-Z], or [0-9], there were spaces, symbols, and a plethora or backspaces and deletes. On a frequent number of the logins, there were instances of a key being pressed repeatedly, a seemingly unusual number of times. Often these were backspace or delete keys. The repeat rate function of the computer operating system repeated and "typed" each keystroke multiple times. The computer logins, or usernames and passwords, were recorded from the data entered in a login window similar to Figure 3.

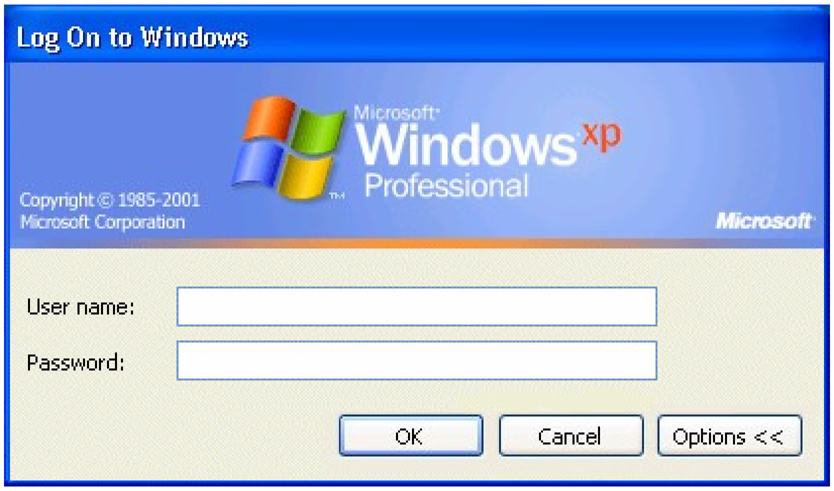

Log On to Windows
Microsoft Windows xp Professional
Copyright © 1985-2001 Microsoft Corporation
Microsoft
User name:
Password:
OK
Cancel
Options <<

Figure 3. Windows Logon screen.

The usernames were assigned to the subjects. Each username was assigned as the subject's first name. To illustrate, some of the names were Terry, Joe, or Lauretta. The passwords were assigned the last for digits of their home phone number. To illustrate, some of the passwords were 2733, 6338, and 0983. The purpose for assigning these was two fold. Primarily these were chosen not to ensure security purposes, as minimal as they may have been, but for their simplicity. Secondarily, they were chosen and enforced in order to provide a type of control measure for this study. Because they were simple, static, and required to be entered before each computing session; they provided data as a steady, constant control activity which can be used in research. These logins are the data in this study, and later research will certainly make use of this data as a control as well.

The usernames and passwords are stored in a separate table accessible through the BRP database. They are not allowed to be changed by the subjects so they remain the same throughout the research project. During the data collection course of this research, no lengthy downtimes of the CMS or database occurred which could have confounded the analysis presented herein.

Methods

We used a variety of computer software to write programs to gather, sort, clean display and analyze the data collected through the BRP study.

The MathWorks™, MatLab 2008b software was used to process the computer activity data. This was accomplished by connecting MatLab to the MySQL database and then saving selected data for each user to a local computer. Initially the program code for this project was written in an earlier version of MatLab. We switched to 2008b not only to improve the code for this project, but also to allow for parallel and future research to make use of a standard platform. Changes in MatLab 2008b include a greater focus on object oriented programming which allowed for rapid algorithm development. This enabled code and data to be more easily transferred between other researchers to improve the workflow of this project. Matlab's strong ability to display data graphically greatly assisted throughout all phases of this research.

I also used R version 2.7.x, which is a free software environment for statistical computing and graphics. [35] After MatLab was used to analyze and process the data, R was used for additional statistical computation and graphical data representation. The capabilities of MatLab and R overlap, however they are primarily built for separate tasks. R is a powerful open source product which has some unique packages. We used this free open source software when performing statistical analysis of the data in this project. R's Lattice package provided the ability to display data from multiple users in a convenient, easily viewable format. One plot per user was graphed while common elements were applied to all plots easily viewed as a matrix of plots. (Figure 4) Figure 4 compares 13 subjects and illustrates time between keystrokes on the x-axis, and density distribution on the y-axis.

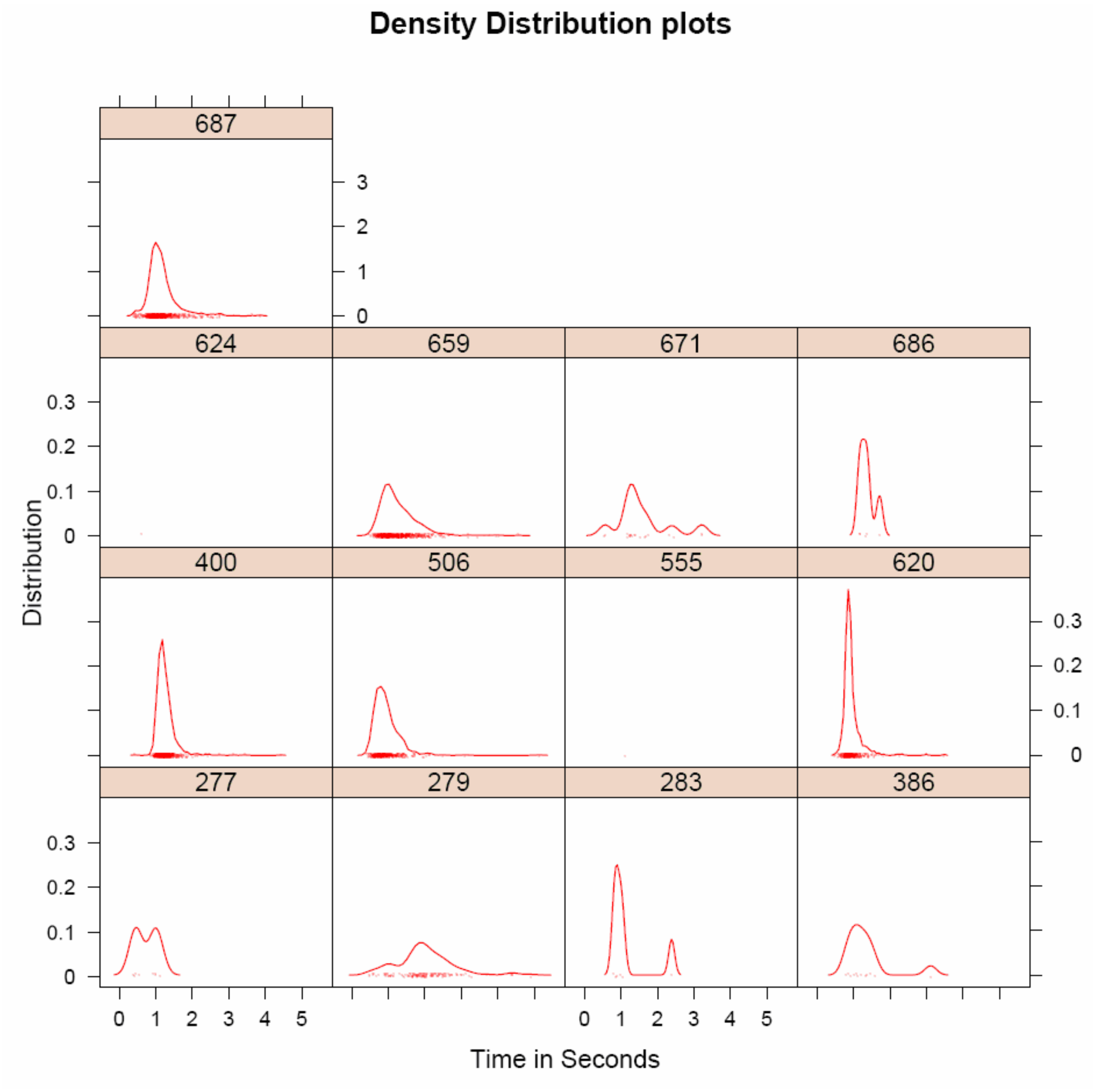


Figure 4. Example of R's Lattice package with a Density Distribution Plot of multiple subjects.

A variety of other graphing tools were used during this research. With these tools we were able to clean the data and present GiB's worth of data in an understandable concise format. We began with a database already containing data. We queried KCKeyData for specific information using SQL. We then chose to search through this data based on user.

After we had data for each user, we would search the data for strings of keystrokes which mapped to the users correct login and passwords. We removed errors from the database to clean the data. There were records attributed to users 0, and -1. These are not real user ids, so they were excluded. There were not a significant number of these which would throw off later analysis. To sort through the data and extract clean information we limited our search to those logins which were typed correctly the first time. This narrowed down our data considerably, enabling us to then move on to later analysis phases of the research project.

Code

The **main.m** code (see appendix) was the parent function for the data gathering, cleaning, and initial analysis functions. Initially, MatLab queried the database to get a list of all users who had data in the database. **db.m** was used to connect to the database. This list was gathered, and then the incorrect user Id's were cleared by **getSubjectID.m**. MatLab would then iterate through that list, running the program on each users data pulled from the database. The user's usernames and ID's were gathered by **userIDpass.m**, and then **sqlGetAll.m** would get data from the database one user at a time.

Then **makeStrings.m** would create strings of correct logins composed of usernames and password to search for in the data **sqlGetAll.m** collected. The function **sql2Vector.m** found the location of all correct matches in the data, and **searchVector2.m** would begin to compute the initial times and deltas between each keystroke. This function would also cut out all logins which took longer than 10 seconds. This cut off was reasoned to be acceptable after viewing the raw data. As illustrated by Figure 5, very few of the correct logins took longer than 10 seconds. The analysis below displays the cumulative deltas between keystrokes for a username of length 7. The data was viewed this way to determine if the deltas between each keystroke appeared normal over time.

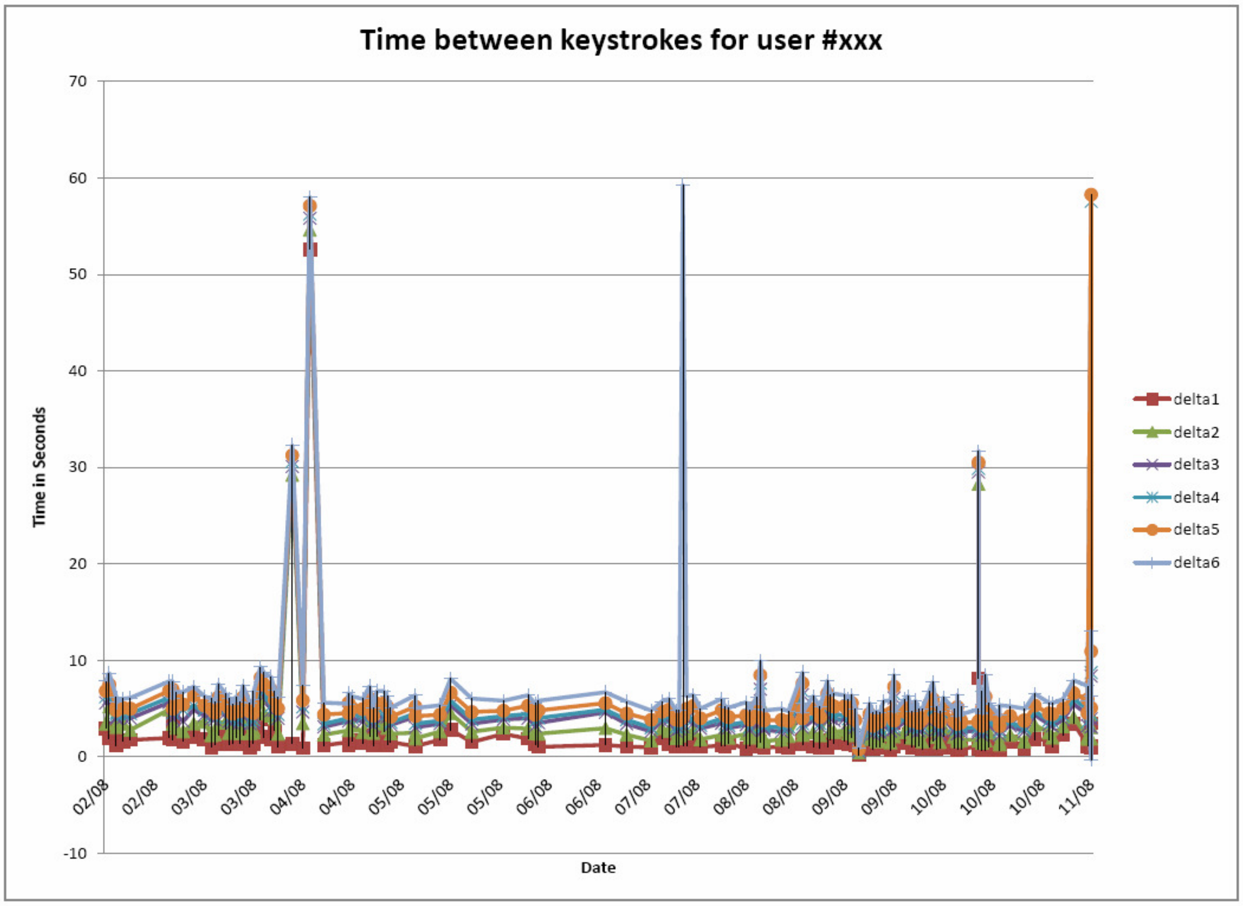


Figure 5. Cumulative time for keystroke deltas in the user name over time for a single user .

The function **saveUser2Matrix.m** saved data in memory to the hard disk. Overall, the **main.m** program took several hours to run, gather data, calculate the time information, and then save it to disk.

Next, a stand alone function **UserPassMatching.m**, was used to query the database and return the usernames where there was a repeating letter, or the passwords where there was a repeating number. This was done in MatLab through the use of regular expressions.

Thirdly, **match.m**, loaded the data previously saved to disk by MatLab. This selected the data for users with a repeat letter in either a username or a password. The data was then saved back to disk.

Next, **main.m**, initialized the connection to the database with **db.m**, and then executed **Kb_Anal01ver2.m**. This function loaded in data from previous code to be analyzed and graphed. Throughout the data analysis process, this function called several other subfunctions. To help with computations of the principal components of each user's login data, **movestat.m** was called to compute the estimate of the central tendencies. Function **getSubjectOADC.m**, was called to give the BRP subject ID, and return the OADC subject ID so the finger tapping data could be called by **getFttTimes.m** from a local file previously received from the OADC. Once this data was loaded in to memory in MatLab, **RawFtt2IntraStrokeAvgMS.m** was called to compute the finger tapping test score to an average time in milliseconds. The function **plot_lines.m** was called to plot the finger tapping score as a vertical line on the probability density function graphs. Finally, R was used to run a linear regression and plot the results.

The theoretical framework and workflow for the finger tapping motion can be diagramed as in Figure 6. This can correspond to either the finger tapping test, or to typing on a computer keyboard. Initially, There is an intentional cognitive command to tap a button. When this is a username, or a familiar string like the subjects first name, there is a search in memory, then a search of the visual field. This then feeds into a feedback loop based where motor control triggers a physical action, or a movement of the finger. The brains sensory processes, touch, sound, and vision return feedback to the motor control system to complete the task. (Figure 6) [19]

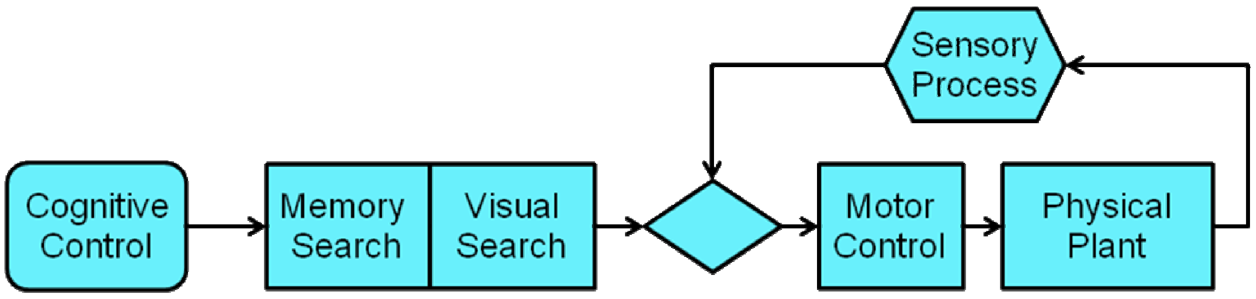


Figure 6. Theoretical Framework for the Finger Tapping Test.

Now that the psychomotor framework has been laid out, we could begin preliminary analysis of keystrokes. We extracted the data from the correct login episodes. The code for following calculations and graphs is located in **Kb_Anal01ver2.m**.

In the following figures, capitalizations is not important, they are just implemented in this graph to easily identify the initial letter of the name. The usernames were all lowercase and thus never correct if a user attempted to capitalize any letter of their username.

We cleaned the raw data to extract the correct login sequences. From there, we computed the initial time of login, and then the deltas between the keystrokes. E.g. Time K1, Delta1-2, Delta2-3, Delt3-n… To illustrated, if a username was “smith”, then we calculated 5 data points. The time of the first ‘s’, and then the delta, or difference in times, between ‘s’ and ‘m’, ‘m’ and ‘i', ‘i’ and ‘t’, and ‘t’ and ‘h’. We graphed and studied the number of logins per day per user, computed of a sliding window. We viewed distribution of all keystrokes. This includes all keystrokes which were part of correct logins. (Figure 7)

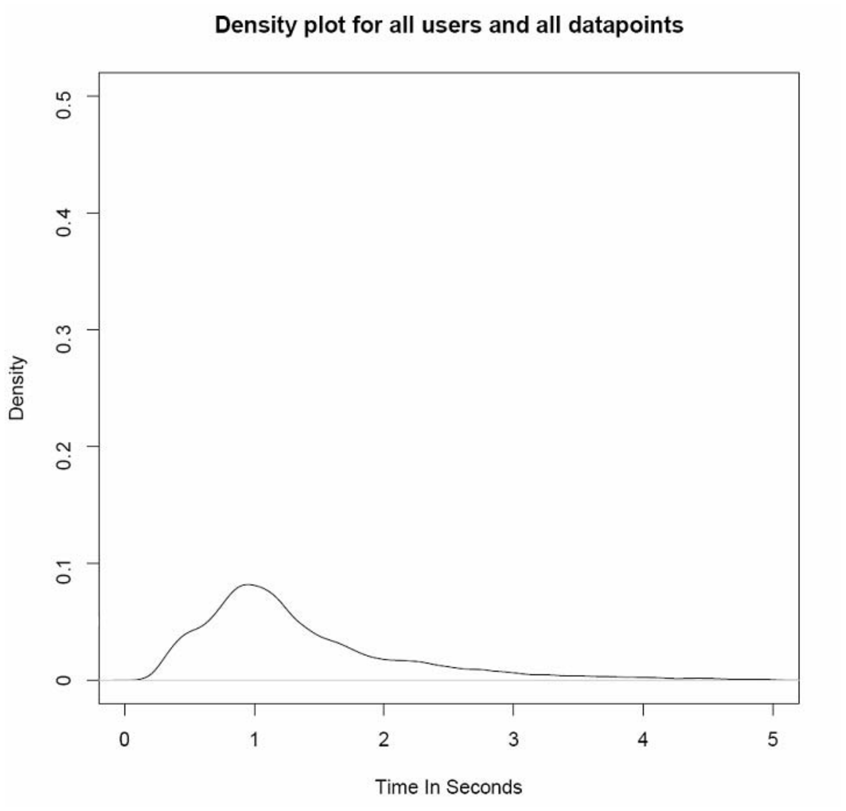


Figure 7. Density distribution plot for all keystrokes.

This rough density distribution illustrates that the majority of the keystrokes were around 1 second. While there is a slight bump between .5 and 1 seconds, this rough overview indicated that there doesn't appear to be any bimodal affect occurring.

The frequency of login was an area of great interest. While the primary concern was to determine the viability of finger tapping vs. keyboard typing, the frequency of login provided needed background information on the data. This data was analyzed per user, and also for all users. Visually we noted certain trends common to many users. There was often a dip in the computer login and usage around December 25th, (note red arrow). Figure 8 displays 20 months of data. Each data point equals the number of logins per day as averaged over that week (7 day period).

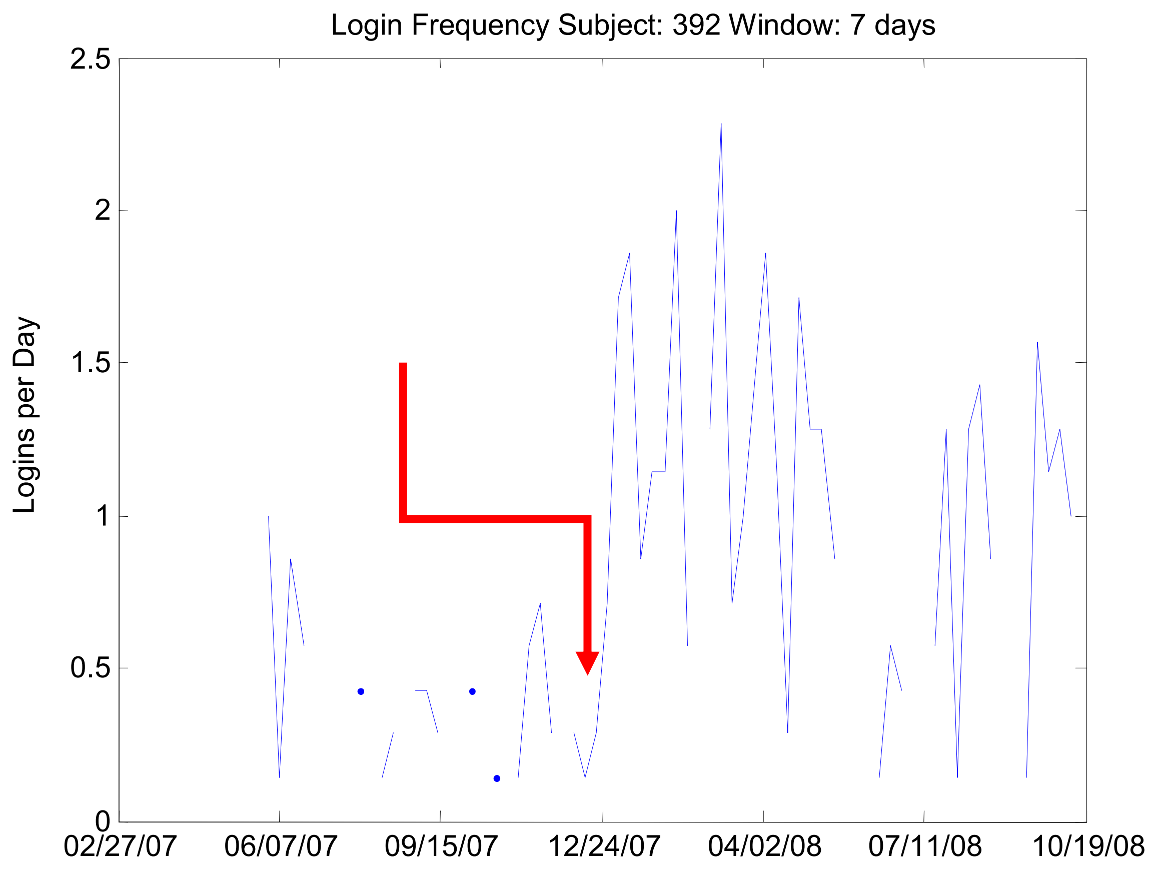


Figure 8. User 392 Login Frequency

After viewing the login frequency, we began to analyze the logins on an individual keystroke level. After computing the deltas between each keystroke, we plotted them per user according to a probability density function. MatLab generated a one graph per user displaying the inter stroke time on the x-axis, and the probability on the y-axis. The y-axis will vary according to each user. The probability density function distribution computes the probability values at points (inter stroke times) on the x-axis, and graphs the probability density function accordingly. Essentially, the taller the height of the function, the more often the inter stroke time was measured at that x-axis value. The inter stroke times between the letters of username “Betty” are graphed in Figure 9.

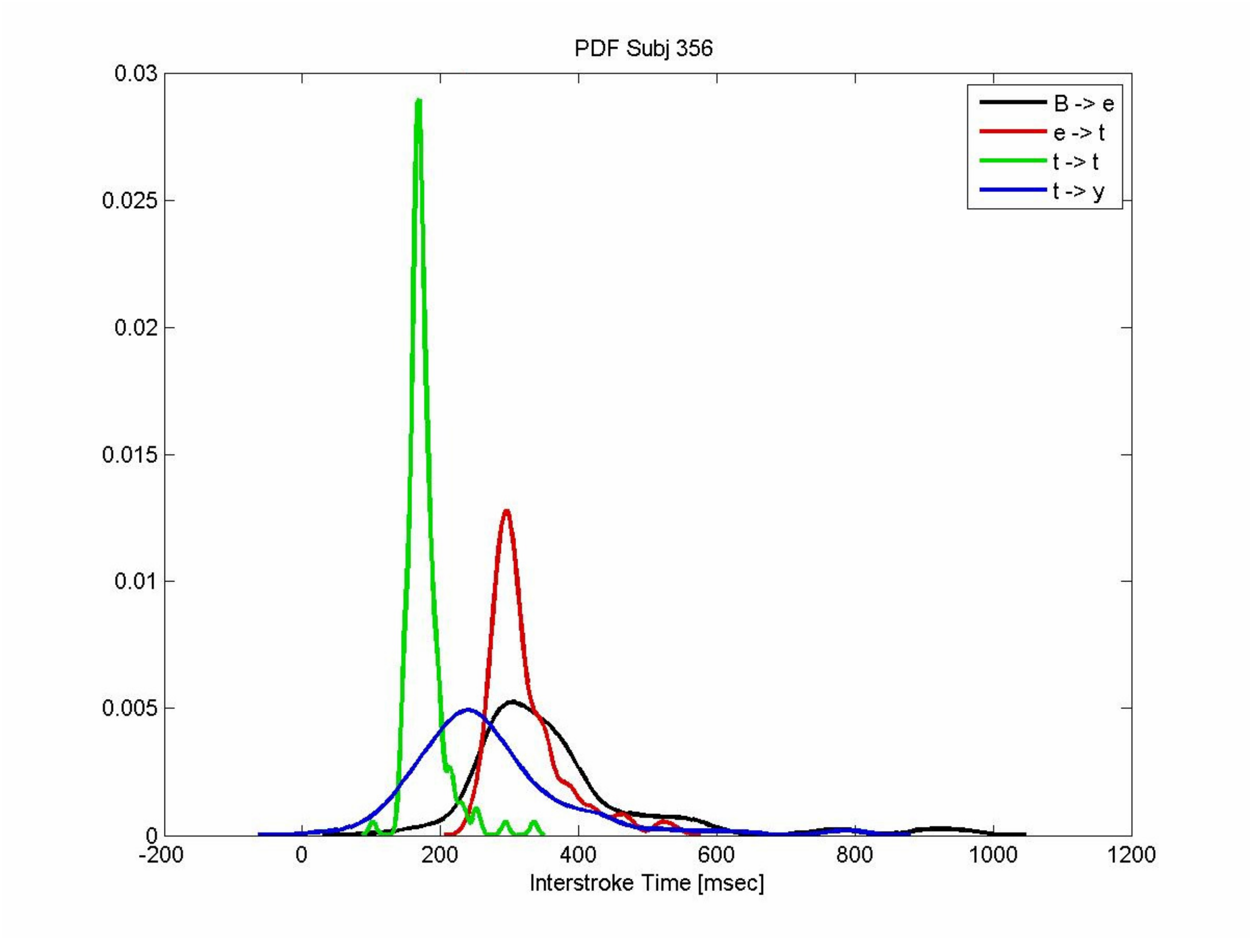


Figure 9. Smoothed Density distribution function of interstroke intervals for username Betty.

To illustrate, the peak of the inter stroke time distribution between ‘t -> t’, in “Betty” is plotted by the green line. In this case, the high peak is formed by the green line. The peak of the green line and the ‘t -> t’ distribution corresponds to 168.5 milliseconds on the x-axis. The height and small width of the peak at 168.5 milliseconds indicates that when Betty repeatedly types the t’s when entering the username, most frequently the time delta between the two keystrokes was 168.5 milliseconds. Betty’s typing speed is not as consistent when typing non-repeating letters, e.g. ‘t -> y’. The curve for ‘t -> y’ is much broader indicating increased variability when typing these two letters.

In order to more fully understand the source of the variability within each login, we performed a principle component analysis via the princomp function in MatLab. For the username “Oregon”, the majority (~45%) of the variance comes from the time in between typing ‘O’ and then ‘r’. Again, capitalizations are not important, just implemented in this graph to easily identify the initial letter of the username. Figure 10 identifies the inter stroke times which are the source of most of the variance from the peak of the distribution in Figure 9. The thin line increasing towards 100% is the cumulative variability explained by each keystroke.

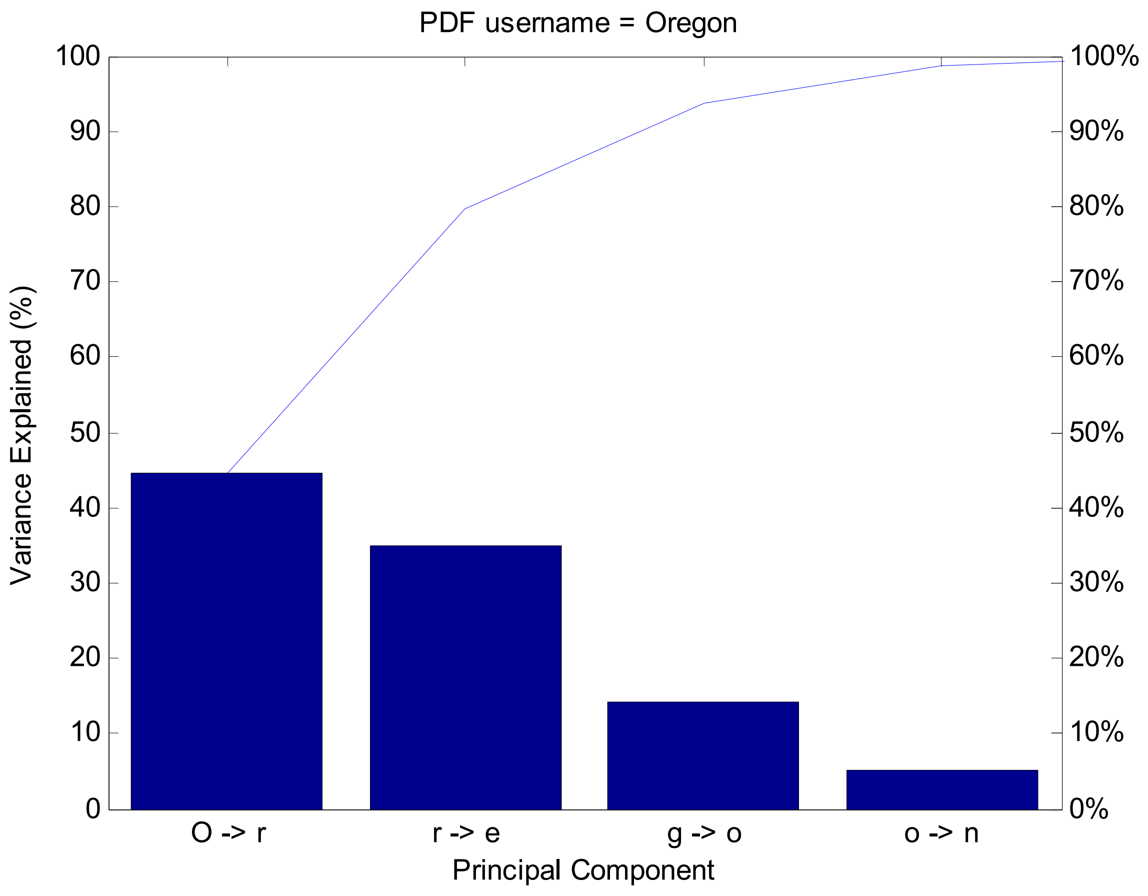


Figure 10. Principle Component Analysis of username “Oregon”.

Analysis of the username's principle components consistently showed that the variability was not usually due to repeated letters. In this case, username 'oregon' doesn't have repeat letters, but the analysis shows interesting information and leaves the door open to future analysis.

We then analyzed the variance of the most significant keystroke in each username. To illustrate, the most significant principle component in the previous figure was due to the variance in 'O -> r'. Analysis of the variance in 'O -> r' is displayed here in Figure 11.

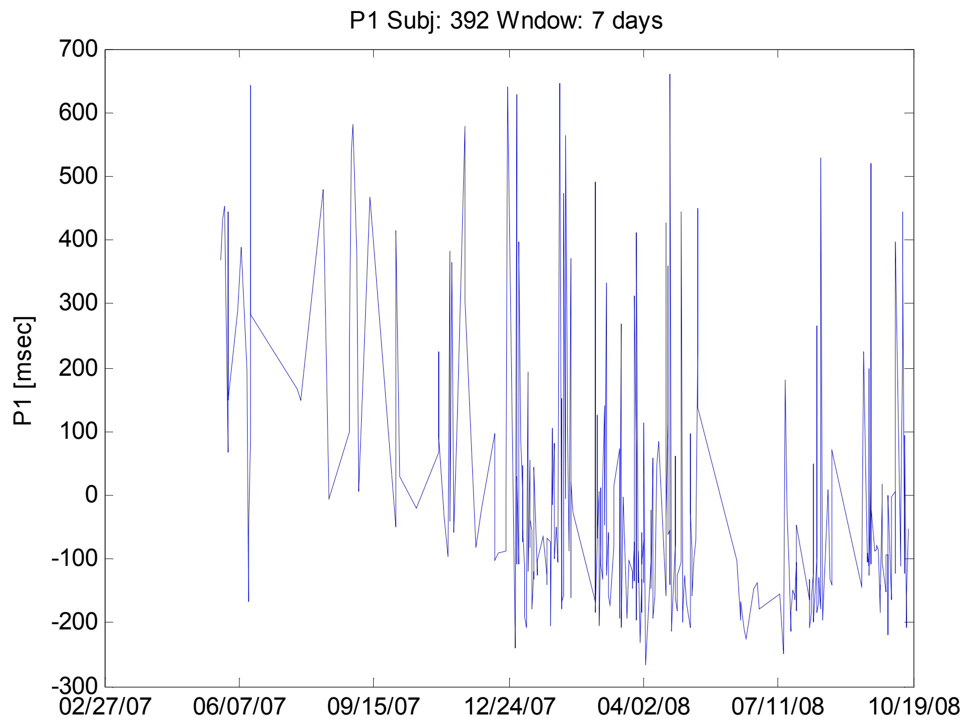


Figure 11. Analysis of most significant principle component.

The analysis here displays the variance over time of typing 'O -> r'. Because we are looking for changes in behavior over time, this is one area which could prove valuable in future research. Because each login can be either faster or slower than the peak of the distribution found in Figure 9, the y-axis of figure x can be either positive or negative.

Returning to the probability density analysis, we compared the speed of typing two repeat letters found in a username. For example, the username “Terry” is one such case. We then gathered the data for these subjects from the NIH Oregon Alzheimer Disease Center (OADC) at Oregon Health & Science University (OHSU). All of the subjects in the BRP study have undergone extensive neuropsychological testing. These results are taken periodically, and are availabe through the OADC. Included in the neuropsych data are the results of the finger tapping test administered to each patient. The finger tapping scores are an average of finger taps in 10 seconds. This number is translated to determine the average time in milliseconds between taps of the finger tapping test. While this finger tapping test score was computed as an average over 5 tests, this does not take into account any variance or variability analysis as with the keyboard typing data.The finger tapping test data from the subjects dominate hand was then plotted on each users PDF according to the x-axis value (Figure 12a,b).

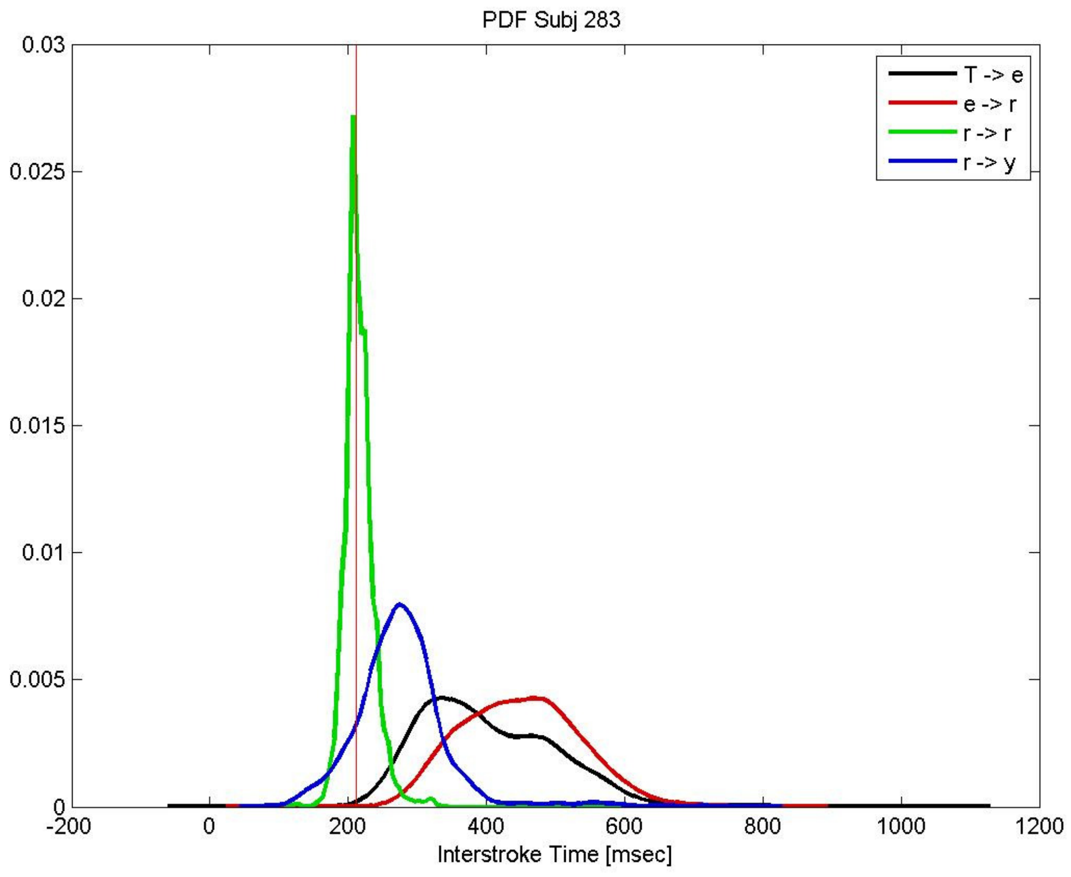


Figure 12a. Finger Tapping Test (red vertical line) with the interstroke interval density function (Username), subject 283.

There are two significant features of this figure. The tallest peak indicates that the inter stroke time for 'r -> r' has a low variability. Also, the peak of the distribution and the average time from the finger tapping test (vertical red line) time line up quite closely. We also see this in subject 731, with the repeating letters accounting for the smallest variability, and thus highest peak. The average time from the finger tapping test also tightly corresponds to the peak of the 't -> t' repeat key stroke Figure 12b.

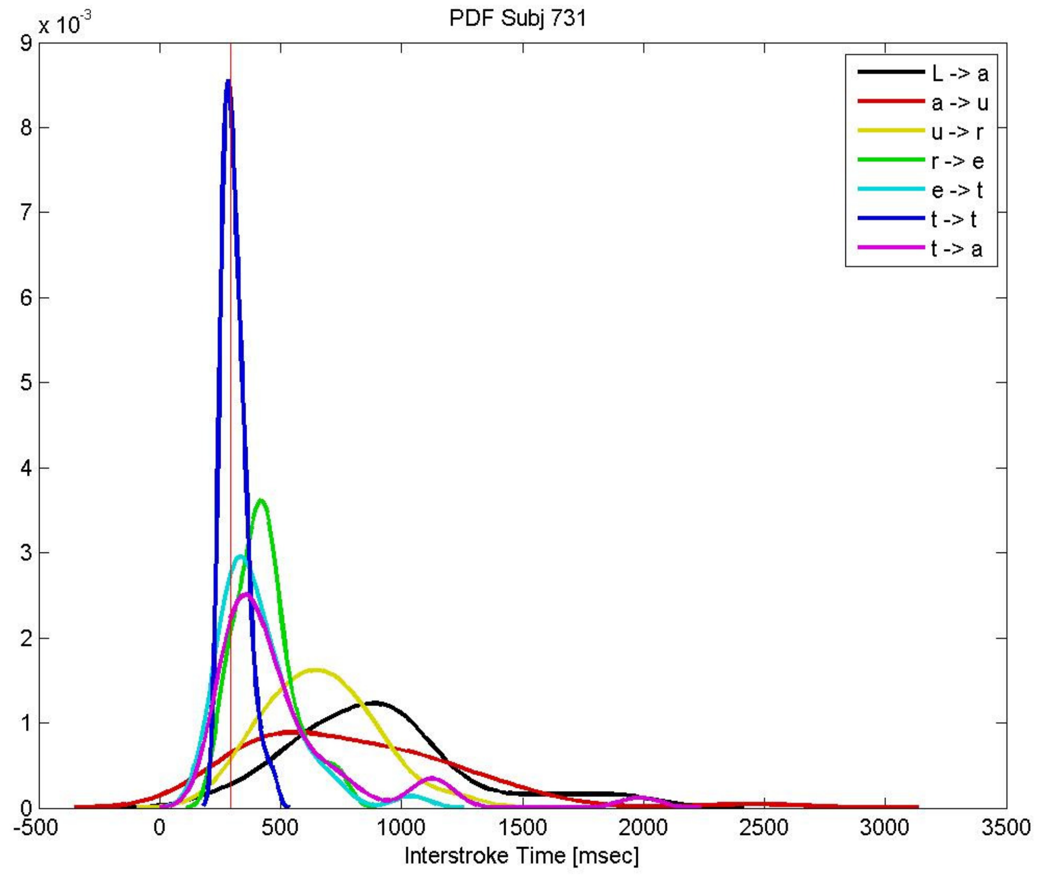


Figure 12b. Figure X. Finger Tapping Test (red vertical line) withinterstroke interval density function (Username), subject 731.

While passwords were not studied as extensively as usernames in this study, this phenomenon was also observed with repeating numbers in a subject's password (Figure 13).

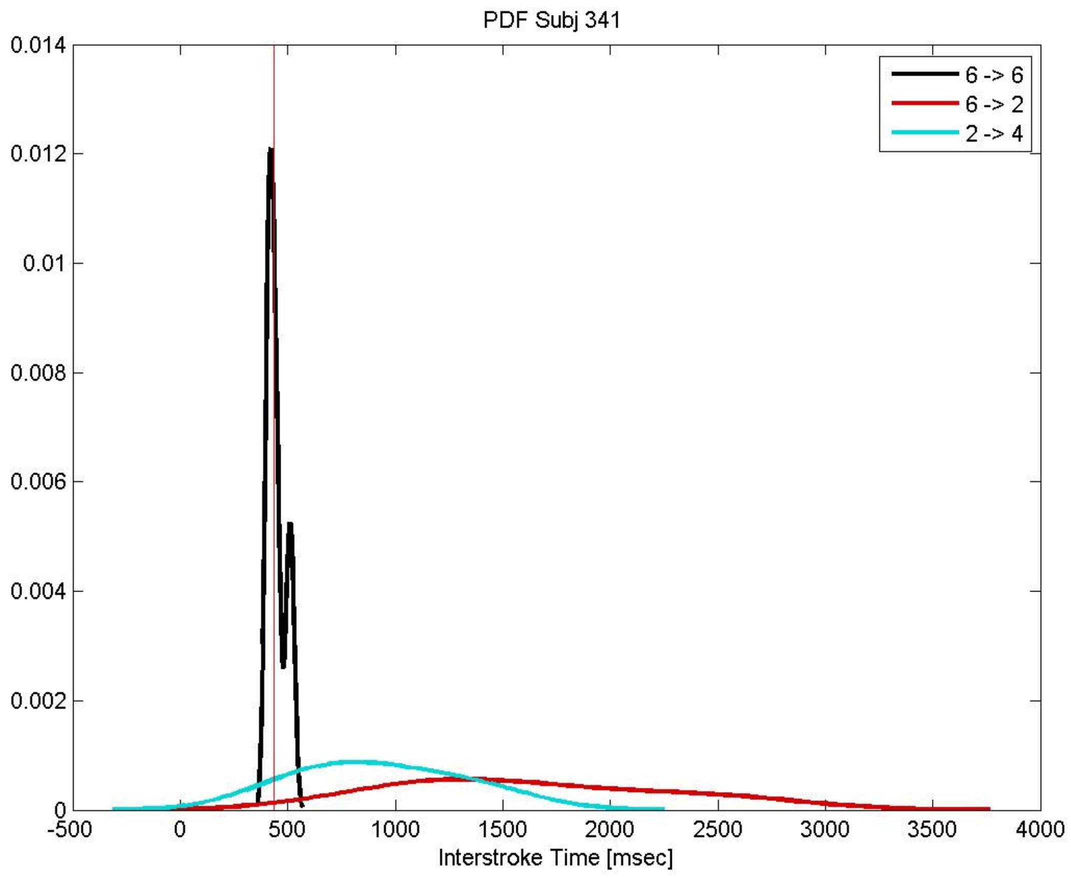


Figure 13. Finger Tapping Test (red vertical line) with Density Function (Password), subject 341.

Continuing our analysis of usernames and finger tapping test scores, we searched for users who had logged into a computer and recorded typing data into the database. Of these 570 users, only 97 of the usernames had repeated letters (e.g. terry). Of the 570 users, not all had been enrolled in the computer aspects of the BRP study, and some who had been enrolled may not have used the computer. We had 196 users with recorded login data. We merged the 196 and the 97. In set theory terms,

A = array
S = set
A $\in$ S, or 97 $\in$ 196 = 18 subjects.

These 18 subjects had both repeat letters in their username, and had data recorded in the KCKeyData table of the database. We then graphed the probability density function and the average finger tapping speed for the dominant hand was plotted on the same graph (Figure 14).

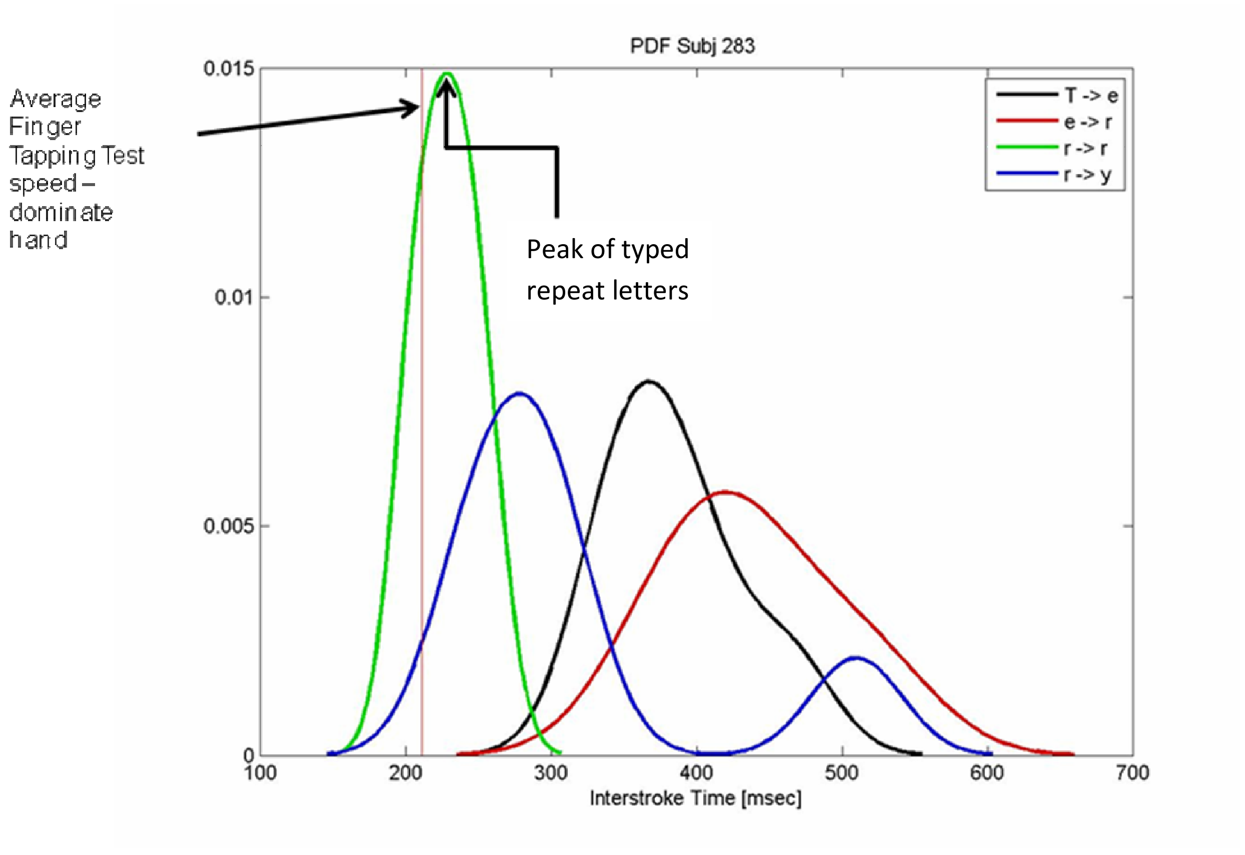


Figure 14. Finger Tapping Test (red vertical line) with Density Functions (Username), subject 283.

Next, we took the peak of the probability distribution function for the typed repeated letter, in this case ‘r -> r’, and found the corresponding x-axis value. We subtracted that value from the finger tapping test x-axis value and took the absolute value. This gave us the distance, or time in milliseconds between the finger tapping test, and the typing speed of repeat letters. We took that value for all 18 users of our subset and graphed the Average typing speed for a repeated letter vs the average finger tapping speed for the dominant hand (Figure 15).

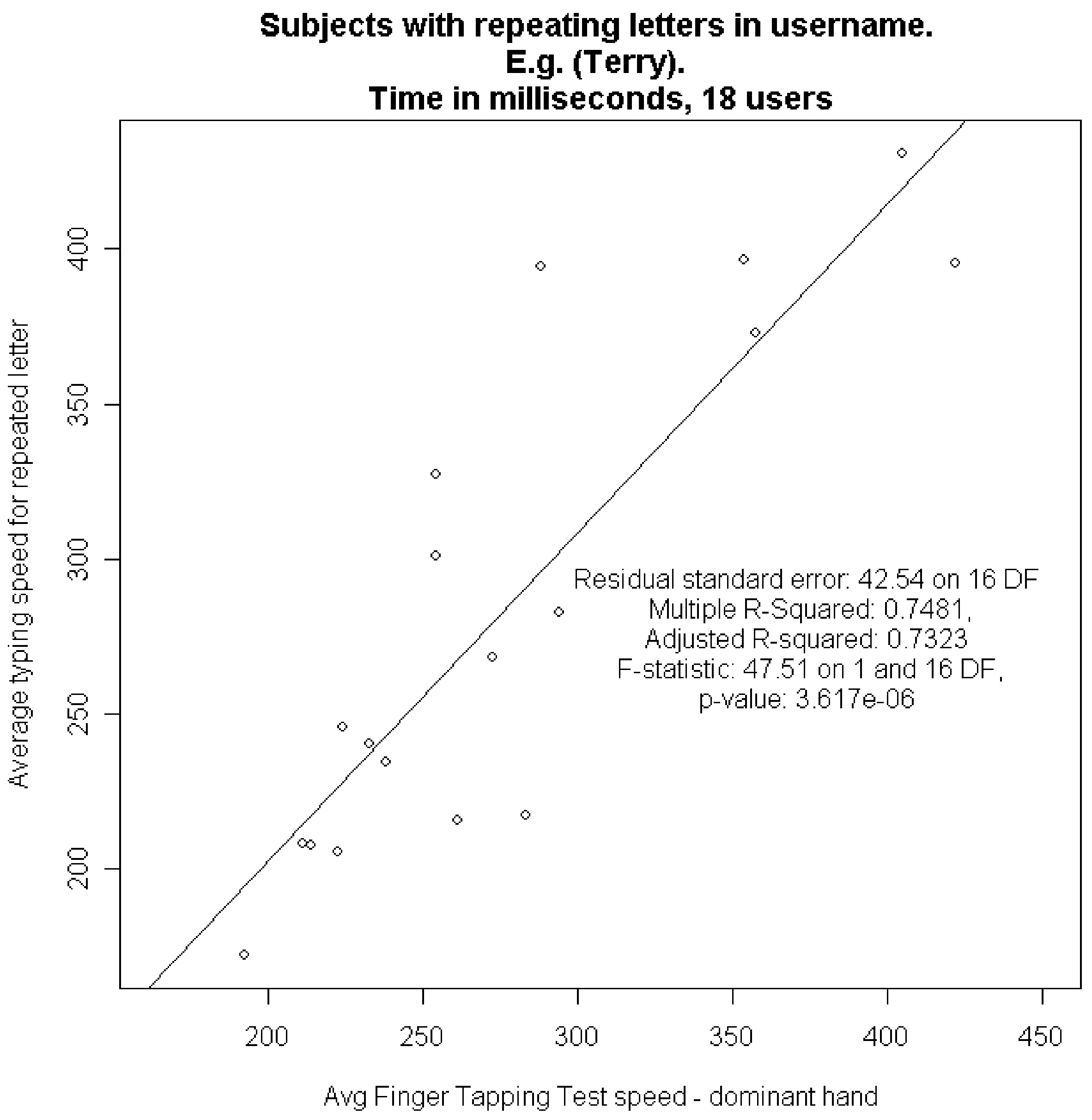
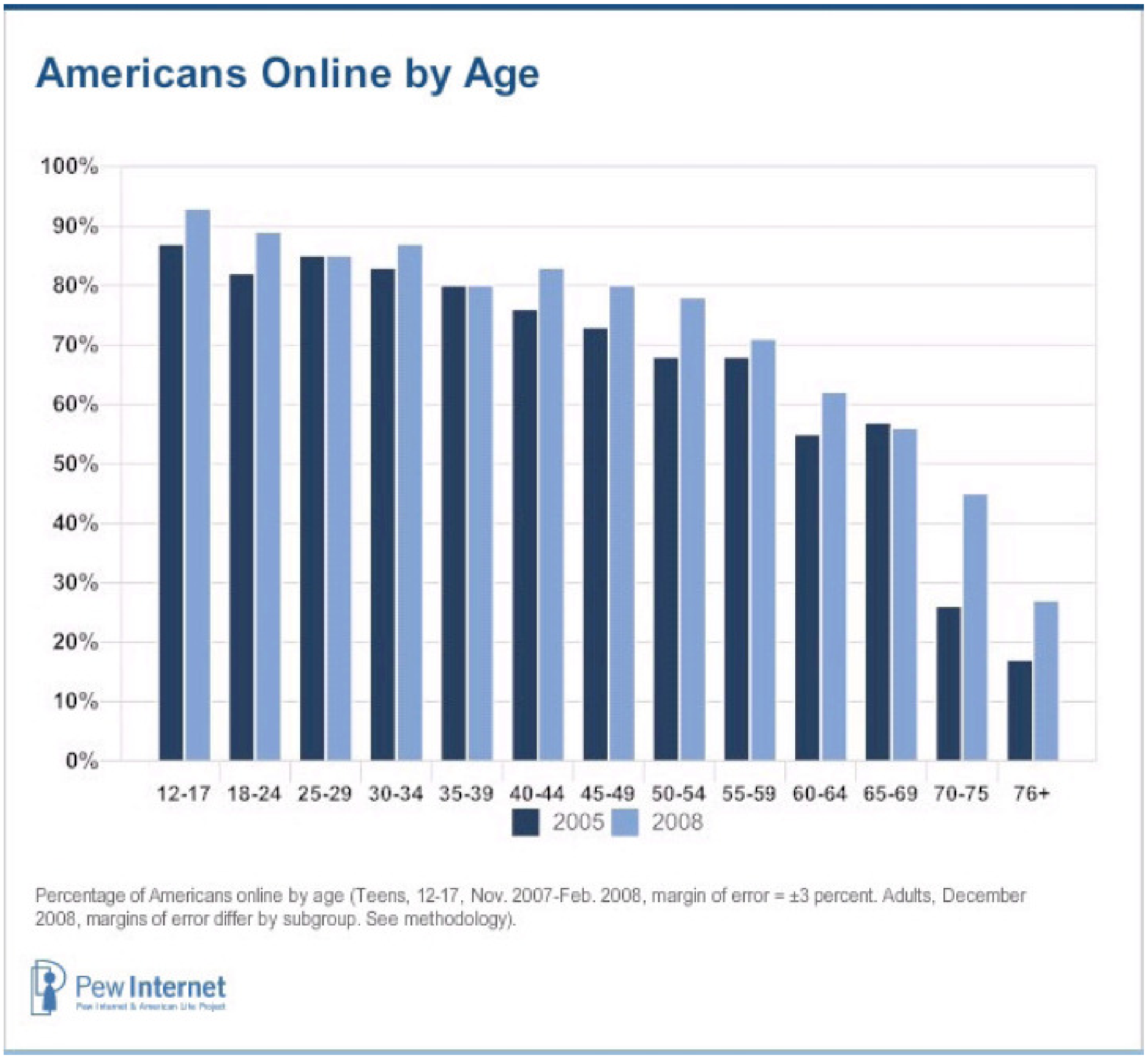


Figure 15. Typing speed (username) vs Finger tapping test.

R was used to perform and graph a linear regression on the data of those 18 users. The results are listed on the figure. The R^2 value is **.7481**, with an F-statistic of 47.51 on 1 and 16 degrees of freedom, with a p-value of $3.617e^{-06}$.

Potential applications

This research undoubtedly lays the ground work for further analyses to be explored. The benefits of unobtrusive monitoring and the potential for monitoring through computer typing data will be explored in the future. According to Pew Internet & American Life Project in 2009, 45% of 70-75 year olds are currently online. [37] Figure 16 displays the number of Americans Online by Age. While the elderly are the group with the lowest percentage of use, this demographic is growing rapidly. As younger Americans age, increased computer use can be expected to remain high as cohort of individuals ages.



Percentage of Americans online by age (Teens, 12-17, Nov. 2007-Feb. 2008, margin of error = ±3 percent. Adults, December 2008, margins of error differ by subgroup. See methodology).

Figure 16. Americans Online by Age [37].

Computers are becoming a staple in American life. As such, they present a unique medium for measuring the users. Because computers require so much intentional cognitive processes interaction to use, they can be employed for actively and passively research opportunities. [23] Behavioral interactions in computers have been passively tracked since the implementation of HTTP browser cookies in 1995. This type of tracking isn't based on cognitive or motor functions, but based on user behavior.

Future research will be able to develop advance algorithms to analyze general keyboard data, not just data limited to computer logins keystrokes.

Discussion

The correlation found here between finger tapping test speed and the typing speed of repeat letters on a computer keyboard holds great potential. The correlation discovered through this research presents an excellent case for the importance of new research in the area of unobtrusive monitoring. There is still significant research needed to investigate unobtrusive monitoring of cognitive decline. The work here has led to new methodologies to accommodate the type of data analyzed in this project. The data collection, organization, and cleaning techniques have been shared in several publications, and used in other studies the usage patterns observed here have led to a better understanding the computing habits of the participants.

There are certainly limitations integral to the dataset. There was a small, but measurable amount of data which was discarded during cleaning. Because we only searched for the correctly typed, correct logins we may have missed other valuable data. For example, if a user typed one extra letter, then typed a backspace to correct the error, and then continued to type the login information correctly, the operating system would register the login as correct and authenticate the user. However, the methodology used here would not recognize and include that login in further analyses.

Conclusion

In this study, data mining algorithms were used to explore the possibility of correlation between the neuropsychological finger tapping test and typing of repeat letters on a computer keyboard. A significant correlation was observed between the finger tapping test and the typing of letters. While this dataset is the first of its kind, we recognize the limitations of the dataset and also the data cleaning methodology. Future work can expand analysis beyond controlled login keystrokes to general computer typing activity. The graphing techniques employed here provide a foundation for future research to analyze similar data efficiently. With further research, this method of unobtrusive monitoring may be used as an indicator of cognitive decline potentially leading to improvements in disease prevention and treatment.

References

1. American Psychological Association, Presidential Task Force on the Assessment of Age-Consistent Memory Decline and Dementia (1998). *Guidelines for the evaluation of dementia and age-related cognitive decline*. Washington, DC: American Psychological Association.

2. Braak, H, U Rub, et al. Cognitive status correlates with neuropathologic state in Parkinson disease. Neurology 2005 Apr; 64: 1404-10.

3. Petersen, R. C., Stevens, J.C., Ganguli, E.G., Tangalos, J.L., DeKosky, S.T. Practice parameter: Early detection of dementia: Mild cognitive impairment (an evidence-based review): Report of the Quality Standards Subcommittee of the American Academy of Neurology. Neurology 2001;56;1133-1142

4. Scarmeas, N, M. Albert, et al. Motor signs predict poor outcomes in Alzheimer disease. Neurology 2005; 64:1696-1703.

5. Miller TP, Tinklenberg JR, Brooks JO, 3rd, Yesavage JA. Cognitive decline in patients with Alzheimer disease: differences in patients with and without extra pyramidal signs. Alzheimer Dis Assoc Discord 1991; 5:251-256.

6. Kraemer HC, Tinklenberg J, Yesavage JA. 'How far' vs 'how fast' in Alzheimer's disease. The question revisited. Arch Neurol 1994; 51:275-279.

7. Chui HC, Lyness SA, Sobel E, Schneider LS. Extra pyramidal signs and psychiatric symptoms predict faster cognitive decline in Alzheimer's disease. Arch Neurol 1994;51:676-681.

8. Morris, JC, Drazner M, Fulling K, Grand EA, Goldring J. Clinical and pathological aspects of parkinsonism in Alzheimer's disease. A role for extranigral factors? Arch Neurol 1989; 46:651-657.

9. Geda, Yonas E., Rosebud O. Roberts et al. Prevalence of neuropsychiatric symptoms in mild cognitive impairment and normal cognitive aging. Arch Gen Psychiatry 2008 Oct; 65 (10): 1193-1198.

10. Petersen RC, Smith GE, Waring SC, Ivnik RJ, Tangalos EG, Kokmen E. Mild cognitive impairment: clinical characterization and outcome [published correction appears in Arch Neurol. 1999;56(6):760]. Arch Neurol. 1999;56(3):303-308.

11. Grundman M, Petersen RC, Ferris SH, Thomas RG, Aisen PS, Bennett DA, Foster NL, Jack CR Jr, Galasko DR, Doody R, Kaye J, Sano M, Mohs R, Gauthier S, Kim HT, Jin S, Schultz AN, Schafer K, Mulnard R, van Dyck CH, Mintzer J, Zamrini EY, Cahn-Weiner D, Thal LJ; Alzheimer's Disease Cooperative Study. Mild cognitive impairment can be distinguished from Alzheimer disease and normal aging for clinical trials. Arch Neurol. 2004;61(1):59-66.

12. Crook T, Bartus R, Ferris S, Whitehouse P, Cohen G, Gershon S. Age-associated memory impairment: proposed diagnostic criteria and measures of clinical change: report of a National Institute of Mental Health Work Group. Dev Neuropsychol. 1986;2:261-276.

13. Reisberg B, Ferris SH, de Leon MJ, Sinaiko E, Franssen E, Kluger A, Mir P, Borenstein J, George AE, Shulman E, Steinberg G, Cohen J. Stage-specific behavioral, cognitive, and in vivo changes in community residing subjects with age associated memory impairment and primary degenerative dementia of the Alzheimer type. Drug Dev Res. 1988;15(2-3):101-114. doi:10.1002/ddr.430150203.

14. Blackford R, LaRue A. Criteria for diagnosing age associated memory impairment. Dev Neuropsychol. 1989;5:295-306.

15. Flicker C, Ferris SH, Reisberg B. Mild cognitive impairment in the elderly: predictors of dementia. Neurology. 1991;41(7):1006-1009.

16. Devanand DP, Folz M, Gorlyn M, Moeller JR, Stern Y. Questionable dementia: clinical course and predictors of outcome. J Am Geriatr Soc. 1997;45(3):321-328.

17. Graham JE, Rockwood K, Beattie BL, Eastwood R, Gauthier S, Tuokko H, McDowell I. Prevalence and severity of cognitive impairment with and without dementia in an elderly population. Lancet. 1997;349(9068):1793-1796.

18. Petersen, R. C., Stevens, J.C., Ganguli, E.G., Tangalos, J.L., DeKosky, S.T. Practice parameter: Early detection of dementia: Mild cognitive impairment (an evidence-based review): Report of the Quality Standards Subcommittee of the American Academy of Neurology. Neurology 2001;56;1133-1142

19. Christianson, Muriel K, Janet M Leathem. Development and standardisation of the computerized finger tapping test: comparison with other finger tapping instruments. New Zealand Journal of Psychology 2004 Jul; 33(2): 44-49.

20. Russell, E.W., Neuringer, C., and Goldstein, G. Assessment of brain damage: A neuropsychological key approach. New York: Wiley – Interscience. 1970.

21. Reitan Neuropsychology Laboratory Inc. Image available from http://www.reitanlabs.com/catalog/product_info.php?cPath=48&products_id=119

22. Jimison, H.; Jessey, N.; McKanna, J.; Zitzelberger, T.; Kaye, J. Monitoring Computer Interactions to Detect Early Cognitive Impairment in Elders. 1st Transdisciplinary Conference on Distributed Diagnosis and Home Healthcare, 2006. D2H2. Volume, Issue , 2-4 April 2006 Page(s):75 – 78

23. Jimison, Holly, Pavel, Misha et al. Unobtrusive monitoring of computer interactions to detect cognitive status in elders. IEEE Transactions on Information Technology in biomedicine 2004 Sep; 8(3):248-252.

24. Jobbagy, Akos, Peter Harcos, et al. Analysis of finger-tapping movement. Journal of Neuroscience Methods 2005; 141:29-39.

25. Scarmeas, N, M. Albert, et al. Motor signs predict poor outcomes in Alzheimer disease. Neurology 2005; 64:1696-1703.

26. T. Hayes, M. Pavel, P. Schallau, AM Adami, "Unobtrusive Monitoring of Health Status in an Aging Population", in Proc. of the 5th Inter. Conf. on Ubiquitous Computing, Oct. 12-15, 2003.

27. Hayes TL, Pavel M, Adami A, Larimer N, Tsay A, Nutt J. Distributed Healthcare: Simultaneous Assessment of Multiple Individuals. IEEE Pervasive Computing. 2007; 6(1): 36-43.

28. Centers for Disease Control and Prevention. Trends in Aging — United States and Worldwide. MMWR 2003;52:102-106.

29. Maciosek MV, Edwards NM, Coffield AB, Flottemesch TJ, Nelson WW, Goodman MJ, Rickey DA, Butani AB, Solberg LI. Priorities among effective clinical preventive services: methods. Am J Prev Med 2006; 31(1):90-96.

30. National Commission on Prevention Priorities. Preventive Care: A National Profile on Use, Disparities, and Health Benefits. Partnership for Prevention, August 2007.

31. National Commission on Prevention Priorities. Data Needed to Assess Use of High-Value Preventive Care: A Brief Report from the National Commission on Prevention Priorities. Partnership for Prevention, August 2007.

32. P. Erickson, R. Wilson, and I. Shannon, Years of Healthy Life. Hyattsville,MD: National Center for Health Statistics, 1995, Statistical Notes, no. 7.

33. L. Boise, D. L. Morgan, J. Kaye, and R. Camicioli, "Delays in the diagnosis of dementia: Perspectives of family caregivers," Amer. J. Alzheimer's Disease, vol. 14, no. 1, pp. 20–26, 1999.

34. P. Glascock and D. M. Kutzik, "Behavioral telemedicine: A new approach to the continuous nonintrusive monitoring of activities of daily living," Telemedicine, vol. 6, no. 1, pp. 33–44, 2000.

35. Software and more information available from: http://www.r-project.org/

36. Figure x. Courtesy of Dr. Misha Pavel.

37. Jones, Sydney., Fox, Susannah. Generations Online in 2009. Pew Internet & American Life Project, January 2009, http://www.pewinternet.org/Reports/2009/Generations-Online-in-2009.aspx, accessed on June 1, 2009. "Americans Online by Age."

38. Okuno, Ryuhei, Masaru Yokoe, et al. Measurement system of finger-tapping contact force for quantitative diagnosis of Parkinson's disease. Proceedings of the 29th Annual International Conference of the IEEE EMBS. August 23026, 2007.

39. Stuss, D. T., I. H. Robertson, et al. 2007 Cognitive rehabilitation in the elderly: a randomized trial to evaluate a new protocol. Jounal of the International Neuropsychological Society 2007; 13:120-131.

40. Richards, Marcus, Raakov Stern, et al. Relationships between extrapyramidal signs and cognitive fnction in a community-dwelling cohort of patients with Parkinson's Disease and normal elderly individuals. Ann Neurol 1993; 33:267-274.

41. Belleville, Sylvie, Louis Behrer, et al. Task switching capacities in persons with Alzheimer's disease and mild cognitive impairment. Neuropsychologia 2008; 46:2225-2233.

42. Maciosek MV, Edwards NM, Coffield AB, Flottemesch TJ, Nelson WW, Goodman MJ, Rickey DA, Butani AB, Solberg LI. Priorities among effective clinical preventive services: methods. Am J Prev Med 2006; 31(1):90-96.

Appendix: Code

Code files listed in order of use for this capstone project.

Matlab®

- main.m
    - db.m
    - fileFixerEnd.m
    - fileFixerStart.m
    - getSubjectID.m
    - makeStrings.m
    - saveUser2Matrix.m
    - searchVector2.m
    - sql2Vector.m
    - sqlGetAll.m
    - userIDpass.m
- UserPassMatching.m
    - db.m
- match.m
- main.m
    - db.m
    - Kb_Anal01ver2.m
        - movestat.m
        - mysql
        - getSubjectOADC
        - getFttTimes
        - RawFtt2IntraStrokeAvgMS
        - plot_lines.m

R: The R Project for Statistical Computing

- graphing scripts

Main.m

```
% Main.m
% This is the parent script for all of the computation for this
% research project.
% Code written by Edward VanBaak, 2008-2009.
% Oregon Health and Science University.
% vanbaake@ohsu.edu

% Clears all variables in the Matlab memory/workspace.
clear all

% Begin Matlab's internal timer.
tic

% Specify the output display format for the data.
format long eng

%Specify the path to the database toolbox files on the local machine.
path = 'C:\Program Files\MATLAB\R2007a\toolbox\mySQLMatlab';

% Connect to the database
db();

% Deletes old temp files and recreates needed blank file structure to be used
later in this script.
fileFixerStart();

%SQL query to get list of subjectId's and then remove, NaN, 0, and -1's.
%Save it in array subIDList.
[subIDList] = getSubjectID(); %returns subIDList

% Initialize variables
Ac = length(subIDList);
ResultsAll = zeros(Ac, 2);
ResultsAll2 = zeros(Ac, 2);
userListLong = [];

% This loop iterates through each user ID and performs the follow code for
% each user ID.
for j=1:(Ac)
    userID = subIDList(j);
    allDeltas = zeros(1,2);

    % SQL command to get usernames and passwords from database.
    userIDpass();

    % Get all data per user. Returns variable keyStrokes.
    sqlGetAll();

    % Creates the strings (correct logins) to search for in keyStrokes.
    makeStrings();

    % Returns the locations of the strings found the keystroke data.
    sql2Vector();

    % This file searches for correct logins, computes the time between each
    % keystroke and total time to log in If the total login is over 10
seconds,
    % it gets thrown out. The data is then saved in various places to be
    % analyzed later.
    searchVector2();

    % Saves userdata as a matrix variable
    saveUser2Matrix();

    %Display program progress
    sprintf('User %s complete; program %3.3f%%
complete.',userIDstring,(100*j/Ac))
end

% Find the unique user id's and store them as variable userList.
userList = unique(userListLong);

% Record number of correct logins per user to file.
dlmwrite('./test/numOfcorrectLoginsperUser.csv', FinalResults);
disp 'done with program'

% Saves workspace variables starting with 'user'.
save('evbdataTEST' , 'user*')

% Clears the data structure used to store data during runtime and
% copies a new blank data structure to be used during the next instance.
fileFixerEnd();

%Close the connection to the BRP database
mysql('close');

%Remove path to the mysql toolbox.
rmpath(path);

% End internal counter, display results.
toc
```

db.m

```
%db.m
%Clear all variables in matlab memory.
% clear all

%Create the connection to the database toolbox.
path = 'C:\Program Files\MATLAB\R2007a\toolbox\mySQLMatlab';

%Add the connection to the database toolbox.
addpath(path);

%initialize db login variables
Database = ('casper.bme.ogi.edu');
Username = ('vanbaake');
Password = ('D4f1b&');

%Execute the connection to the DB.
mysql('open', Database, Username, Password);

%Set the current database to use BRP database
mysql('use BRP');


% Display information about the connection and the server.
% Return    0  if connection is open and functioning
% 1  if connection is closed
% 2  if should be open but we cannot ping the server
mysql('status');
```

fileFixerStart.m

```
% fileFixerStart.m
% This file initializes the correct file structure needed to run the main()
% functions (and its dependancies). It removes files from previous
% occurances of the code, and clears the slate for the current process.

% Delete evbdataTEST in local directory - it exists from last time the
% program ran.
delete('./evbdataTEST.mat')

% Delete all of the *.csv data files in the ./test directory
delete('./test/*.csv')

% Delete all of the *.csv Delta data files written in ./test/Delta/
delete('./test/Delta/*.csv')

% Copies a blank .mat file from local directory to local (essentially
% renaming the file but not deleting the original file
copyfile('./evbdataBLANK.mat','./evbdata.mat')
```

getSubjectID.m

```
% getSubjectID.m
% This function queries the database to get all of the subject ID's and
% then returns them as a matrix 'subIDList'.
function  [subIDList] = getSubjectID()

%SQL query to get list of subjectId's
SQLQuery1 = ('SELECT DISTINCT subjectID FROM KCKeyData ORDER BY subjectID
DESC;');

% Submit Query, clear variable, and assign len
subIDList = mysql(SQLQuery1);
clear SQLQuery1;

% Remove the NaN, 0, and -1's from the list of subject ID's.
for i=1:(length(subIDList))
    An = isnan(subIDList(i));
    if (An == 1)
         subIDList(i) = 0;
    end

    if (subIDList(i) == -1)
         subIDList(i) = 0;
    end
end

sort(subIDList, 'descend');

% Returns sorted subIDList as array.
while (subIDList(end) == 0)
    subIDList(end) = [];
end
```

userIDpass.m

```
% UserIDpass.m
% SQL command to get usernames and passwords from subjects_new table.
% Give A and B arrays with info.
% A contains User ID and Password
% B contains string's of names

% userName1 is a matrix containing:
% userName2 is a matrix containing:
% userName3 is a matrix containing:

% Open connection to mySQL toolbox and choose which database to use.
mysql('use subjects_new');

% Set userIDstring to be used for creating string mySQl statement queries
userIDstring = num2str(userID);

% Construct mySQL query
SQLQuery2 = ['SELECT * FROM login_info where idx = ' userIDstring ';'];

% Execute query and store results of mySQL query
% a = user ID
% b = first name string
% c = password / last 4 of phone number
% e.g. [479, John, 6459]
[a,b,c] = mysql(SQLQuery2);

% Set username string to be used later in program
userName1 = char(b);
C = str2num(char(c));

% Set variables to be used later in program
A = [a C];
userName2 = [a C];
userName3 = c;
```

sqlGetAll.m

```
% sqlGetAll.m
% Queries the database for login typing data for userID.
% The data is extracted via an SQL query by user. The data is collected per
% keystroke, and these elements are gathered: subjectID, KeyID, EventDate,
% EventTime, and MSec.

list = ['subjectID, KeyID, EventDate, EventTime, MSec'];

% Specifies that the BRP database should be used.
mysql('use BRP');


SQLQuery10 = ['SELECT ' list ' FROM KCKeyData WHERE subjectID = '
userIDstring ' ORDER BY EventDate ASC, EventTime ASC, MSec ASC;'];
[aa, bb, cc, dd, ee] =  mysql(SQLQuery10);

% Saves data to be analyzed later in program.
keyStrokes = [aa, bb, cc, dd, ee];
% Clear variables from working memory.
clear [aa, bb, cc, dd, ee];
```

makeStrings.m

```
% makeStrings.m
% This file creates the strings to search for in the keystroke data.
% The searching will be performed later in the code by this function:
% k = strfind(str, pattern)
% Two strings will be created.
% String1 = [TERRY1234]
% String2 = [TERRY$1234]

% Format of string one:
% UxPx

% Use variable userName3 from userIDpass()
% e.g.(1234)
password2 = userName3;

% If a keystroke was a 'tab' (ascii code = 9), then change the character to
% a $ (ascii code 36).
N = size(keyStrokes);
for i=1:(N(1,1))
    if (keyStrokes(i,2) == 9)
        keyStrokes(i,2) = 36;
    end
end

% tab
tab = ['$'];

% string one:
% UsernamePassword
% Initialized the strings to be used in later code.
% e.g. s1 = [TERRY 1234];
s1 = [char(upper(userName1)) char(password2)];

% e.g. es1 = [TERRY]
es1 = [char(upper(userName1))];

% e.g. passwordString1 = [1234]
passwordString1 = [char(password2)];

% string two:
% UsernameTabPassword
% This string will also find those logins which used a tab/enter/mouse click
% after entering the password.
% e.g. s2 = [TERRY$1234]
s2 = [char(upper(userName1)) char(tab) char(password2)];
```

sql2Vector.m

```
% sql2Vector.m
% Returns the locations of the strings found the keystroke data which
% matched to either s1 or s2.
% keyStrokes = ['subjectID, KeyID, EventDate, EventTime, MSec'];
results  = strfind(keyStrokes(:,2)', s1);
results2 = strfind(keyStrokes(:,2)', s2);

% Records the number of results (correct logins) found
ResultsAll(j,:) = size(results);
ResultsAll2(j,:) = size(results2);

ResultsAll(j,1) =  A(1,1);

% Combine the 2 lists and create the FinalResults variable.
Results3 = ResultsAll(:,2) + ResultsAll2(:,2);
FinalResults = [ResultsAll(:,1) Results3];
```

searchVector2.m

This code supports both computations of username and password data. Changes in commenting determines which dataset the code will compute.

```
% searchVector2.m
% This file searches for correct logins, computes the time between each
% keystroke and total time to log in If the total login is over 10 seconds,
% it gets thrown out. The data is then saved in various places to be
% analyzed later.

% This searches for strings s1 & s2 in keyStrokes(:,2)'.
results  = strfind(keyStrokes(:,2)', s1);
results2  = strfind(keyStrokes(:,2)', s2);

% Store the results as a differently named array.
newResults1 = results;
newResults2 = results2;

% Sorts and combines the results into one array.
entireResults = sort([newResults1'; newResults2']);

% Determine the number of results.
ResLen = (length(entireResults'));

% Initialize empty matrices
avgMatrix = [];
newDeltas = [];
deltaMatrix = [];


% Iterates through the list of results (strings which matched to s1 or s2).
for k=1:ResLen
%   Initialize empty matrices
    AddAll = [];
%   Location of login in keyStrokes matrix.
    startOfRange = entireResults(k,1);

    % Iterates through keyStrokes matrix to extract (by iteration) to
    % get matrix of correct logins extracted into AddAll.
    for m=0:(length(userName1)-1)
        T = m + startOfRange;
        add1 = keyStrokes(T,:);
        AddAll = [AddAll; add1];
    end
    % Find the time the login started.
    timeGuess = [keyStrokes(startOfRange,3) keyStrokes(startOfRange,4)
keyStrokes(startOfRange,5)];

    % Find the length of login.
    UMM = (size(AddAll)-1);
    % Iterate through the login keystrokes and calculate the length of time
    % between each keystroke (n, n+1, n+...)
    for ITT = 1:(UMM(1,1))
```

```
        unTime = 0;
        time1 = AddAll(1,:);
        time2 = AddAll(ITT+1,:);

        % Calculate the milliseconds.
        ms1 = time1(1,5);
        ms2 = time2(1,5);
        msdiff = ms2 - ms1;

        % Calculate the h:m:s
        h1 = time1(1,4);
        h2 = time2(1,4);
        hourdiff = h2 - h1;

        %format hour approriatly
        clockFormat = datestr(hourdiff, 13);


        % calculate the cumulative time between keystroke n, and n+1, then
        % append the time (in milliseconds) to avgMatrix.
        timeSecondTensPlace = str2num(clockFormat(1,end-1));
        timeSecondOnesPlace = str2num(clockFormat(1,end));
        at = ((timeSecondTensPlace * 10) * 1000);
        bt = (timeSecondOnesPlace * 1000);
        totTimePer =  [(at + bt + msdiff)];
        avgMatrix = [avgMatrix; i totTimePer];

        % This will give the initime, and then the rest of the times
        % cumulativly e.g [1;2;3;4;5], not [1;1;1;1;1]
        deltaMatrix = [deltaMatrix; i avgMatrix(end,2)];

        % Calculate values to be analyzed by later code.
        newMean = [userID mean(avgMatrix(:,2)) timeGuess];
        newDeltas = [newDeltas; timeGuess avgMatrix(end,2)];
        initTime = (timeGuess(1,1) + timeGuess(1,2) + (timeGuess(1,3) *
(1/86400/1000)));
        newDeltasCorrect = [initTime; deltaMatrix(:,2)];

        % If the login exceeds 10 seconds, the breakKey flag gets set.
        for i=2:length(newDeltasCorrect)
        % newDeltasCorrect(i);
            breakKey = 0;
            if (newDeltasCorrect(i) >= 10000)
                breakKey = 1;
            end
        end
    end

% If login time is longer than 10 seconds, the login is deleted, and the
% next correct login in computed.
if (breakKey == 1)
    disp '10 second login cut off reached'
    avgMatrix = zeros(1,2);
    newDeltas = [];
    deltaMatrix = [];
    continue
end
```

```
% Writes to disk the time delta's between keystrokes to a csv file named per
user.
fileNameTimeDelta = ['./test/Delta/TimeDelta' userIDstring '.csv'];
dlmwrite(fileNameTimeDelta, newDeltasCorrect', '-append', 'precision',
'%15.7f');

% clear variables for next iteration.
avgMatrix = [];
deltaMatrix = [];
clear allTimeHere;

% Appends and saves the mean time of all logins per user to a csv file.
fileNameTime = ['./test/allusers.csv'];
dlmwrite(fileNameTime, newMean, '-append', 'precision', '%15.7f');

% Appends and saves the mean time of all logins per users to 1 csv file per
% user.
fileNameTime2 = ['./test/' userIDstring '.csv'];
dlmwrite(fileNameTime2, newMean, '-append', 'precision', '%15.7f');

% Saves the data in file to a variable to be used by later code.
% This should produce the code:
% user### = csvread('user###.csv')
s = ['user' userIDstring ' = csvread(''./test/Delta/TimeDelta' userIDstring
'.csv'');'];

% This should evaluate s, creating the variable(s) user###
eval(s);

% If user evaluated was 0, delete user 0's data.
userListLong = [userListLong; userID];
if (userListLong(1,1) == 0)
    userListLong(1,:) = [];
end
% Clear matrix for next iteration.
clear AddAll
end
```

saveUser2Matrix.m

```
% saveUser2Matrix.m
% This file saves userName1 as matrix userName###

% userName### = userName1
s2 = ['userName' userIDstring ' = userName1'';'];
% This should evaluate s, creating the variable(s) user###
eval(s2);
```

fileFixerEnd.m

```
% fileFixerEnd.m
% This file clears the data structure used to store data during runtime and
% copies a new blank data structure to be used during the next instance.
delete('./mishaNew/evbdata.mat')

copyfile('./evbdataTEST.mat','./mishaNew/evbdata.mat')
```

UserPassMatching.m

```
% UserPassMatching.m
% This script queries the database and return the usernames and passwords
% for each user ID where either the username or the password has a
% repeating double letter. (e.g. eddie)

% Clears all variables in the Matlab memory/workspace.
clear all

% Connect to the database
db();

% Specifies correct database to connect to.
mysql('use subjects_new');

% Selects all data from specifed table.
% Returns data in the following format:
% subjectID, 'Firstname', 'password'
% 123, 'Eddie', '4567'
SQLQuery1 = ('SELECT * FROM subjects_new.login_info l;');

% Query database
[idx username password] = mysql(SQLQuery1);

% Close mySQL connection
mysql('close');

% Initialize empty matrices.
newNames = [];
newNamesIds = [];

% Finds the username where username has a double letter.
% e.g. 'Eddie'
for i=1:length(idx)
    [user_A user_B user_C] = regexp(username(i), '(\w)(\1+)', 'match');
    if isempty(user_A{1}) == 0
        newNames = [newNames; idx(i) username(i) password(i)];
        newNamesIds = [newNamesIds; idx(i)];
    else
        continue
    end
end

save newNames
save newNamesIds
disp 'Done with username matching: newNames'

% Initialize empty matrices.
newPasswords = [];
newPassIds = [];

% Finds the usernames where password has a double letter.
for ii=1:length(idx)
    [pass_A pass_B pass_C] = regexp(password(ii), '(\w)(\1+)', 'match');
    if isempty(pass_A{1}) == 0
        newPasswords = [newPasswords; idx(ii) username(ii) password(ii)];
        newPassIds = [newPassIds; idx(ii)];
    else
        continue
    end
end

save newPasswords
save newPassIds
disp 'Done with password matching: newPasswords'
disp 'Done with program'
```

db.m

```
% db.m
% This gets called to initialize a connection to a MySQL database.

%Clear all variables in matlab memory.
clear all

%Create the connection to the database toolbox.
path = 'C:\Program Files\MATLAB\R2007a\toolbox\mySQLMatlab';

%Add the connection to the database toolbox.
addpath(path);

%Initialize these hardcoded login variables
Database = ('casper.bme.ogi.edu');
Username = ('vanbaake');
Password = ('D4f1b&');

%Execute the connection to the DB.
mysql('open', Database, Username, Password);

%Specify which db/table to connect to
mysql('use BRP');


% Display information about the connection and the server.
% Return    0  if connection is open and functioning
% 1  if connection is closed
% 2  if should be open but we cannot ping the server
mysql('status');
```

match.m

```
clear all
% Loads data from main script, which loads a userlist (saved as part of
% evbdata as variable userList)
load('../evbdata.mat', 'userList');
usersWcompdata  = userList;
% from ../matching/UserPassMatching.m
% loads names which have 2 repeat letters in their usernames.
load ('../matching/newNamesIds.mat');
% loads names which have 2 repeat letters in their passwords.
load ('../matching/newPassIds.mat');

% % % % % % %
% USERNAMES
% % % % % % %
% gets list of usernames with repeat letters and compares to users with
% computer data.
[aa bb] =  ismember(newNamesIds, usersWcompdata);
% [aa bb] = ismember(usersWcompdata, doubleLetters);

usersWithDataAndRepeatLetters = [];

 for i=1:length(newNamesIds)
    if aa(i) ==1
        usersWithDataAndRepeatLetters = [usersWithDataAndRepeatLetters;
newNamesIds(i)];
    end
 end

 save usersWithDataAndRepeatLetters;

% % % % % % %
% PASSWORDS
% % % % % % %
% gets list of passwords with repeat letters and compares to users with
% computer data.
[aa bb] =  ismember(newPassIds, usersWcompdata);

usersWithDataAndRepeatNumbers = [];

 for i=1:length(newPassIds)
    if aa(i) ==1
        usersWithDataAndRepeatNumbers = [usersWithDataAndRepeatNumbers;
newPassIds(i)];
    end
 end
 save usersWithDataAndRepeatNumbers;
```

main.m

```
clear all

db();

Kb_Anal01ver2();
```

db.m

```
%db.m
%Clear all variables in matlab memory.
% clear all

%Create the connection to the database toolbox.
path = 'C:\Program Files\MATLAB\R2007a\toolbox\mySQLMatlab';

%Add the connection to the database toolbox.
addpath(path);

%initialize db login variables
Database = ('casper.bme.ogi.edu');
Username = ('vanbaake');
Password = ('D4f1b&');

%Execute the connection to the DB.
mysql('open', Database, Username, Password);

%Set the current database to use BRP database
mysql('use BRP');


% Display information about the connection and the server.
% Return    0  if connection is open and functioning
% 1  if connection is closed
% 2  if should be open but we cannot ping the server
mysql('status');
```

Kb_Anal01ver2.m

```matlab
% Analysis of keystrokes
% Data are in the form of a column for each letter of the login
% This code has been adapted from Misha Pavel, PhD.
tic;

% Load M from evbdata (from previously run code which
% gathered data from the BRP database.
load evbdata;

load('./merge/usersWithDataAndRepeatNumbers.mat',
'usersWithDataAndRepeatLetters');

% subjs = userList'; %used user list from evbdata
subjs = usersWithDataAndRepeatLetters';

% plot? Flag: 0 = no, 1 = yes.
plotFlag = 1;

% this makes a matrix which will hold the data to be written at the end of
% the for loop.
% IdAndFttDataMat = [IdAndFttDataMat; BRPOutput OADCOutput FttNonDom
% FttDom]
IdAndFttDataMatRaw = [];
IdAndFttDataMatMS = [];
UserPeaks = [];
UserPeaksTest = [];
UserPeaksTest2 = [];

for sid=subjs  % sid = subjs(1)  sid = 1;
    userIDstring = num2str(sid);

    M = eval(sprintf('user%d',sid));
    numPoints = length(M);
    [ntr nkeys] = size(M);

%     if there is only 1 login, then skip to next subject ID.
    if (ntr < 3)
        continue
    end

    [traw,ix] = sort(M(:,1));

    M = M(ix,:);
    t = traw - traw(1);
    d = M(:,2:nkeys);

    if plotFlag == 1
       % Raw interdigit times to determine if outliers need to be removed
        for i=2:nkeys
        hist(M(:,i))
% % %      plot(M(1:end,i))
        end
    end
    % --------------  USAGE ANALYSIS ------------------------------
    % Average  number of logins
    w = 7; % Window in days
    wn = 0;
    clear nlogin tlogin
    for ti = w:w:t(end)
        % compute sum
        idx = find((t>(ti-w)) & (t<=ti));
        wn = wn + 1;
        nlogin(wn) = length(idx);
        tlogin(wn) = traw(1)+ti;
    end
    nlogin(find(nlogin==0))= NaN;
% % %     figure
% % %     plot(tlogin, nlogin/w)
% % %     title(sprintf('Login Frequeny Subj: %d Wndow: %d days', sid, w'));
    if plotFlag == 1
            XTicksLabels = datenum(get(gca,'XTick'));
    end

% % %     set(gca,'XTickLabel',datestr(XTicksLabels,2));
% % %     ylabel('Logins per Day');


    % Compute times between successive successful logins
    del = diff(t);
    if plotFlag == 1
        ksdensity(del)
    end
    % If data are in the cummulative difference format
    % Computer inter-stroke times
    d = [d(:,1) diff(d,1,2)];  % shoud use diff(d,1,2) row instead of cols

%     remove negative entries.
    [rn c] = find(d<0);
    [rp c] = find(d>0);
    r = unique(setdiff(rp,rn));
    d = d(r,:);
    t = t(r);
    traw = traw(r);

    % Remove outliers
    dt = sum(d')';
    dlim = prctile(dt,95);
    idx = find(dt<dlim);
    d = d(idx,:);
    t = t(idx);
    traw = traw(idx);

% % %     figure
% % %     plot(t,d)
% % %     corr(d);

% [xhat medhat sdhat] = movestat(d(:,2), t, w};

    [coefs,scores,eignv2,t2] = princomp(d);
```

```
% % %     plot(scores(:,1),scores(:,2),'+')
% % %     xlabel('1st Principal Component');
% % %     ylabel('2nd Principal Component');
% % %     plot(coefs(:,1:2))
    % gname  % to identify points
    percent_explained = 100*eignv2/sum(eignv2);
% % %     pareto(percent_explained)
% % %     title(sprintf(' Subj: %d ', sid));
% % %     xlabel('Principal Component')
% % %     ylabel('Variance Explained (%)')


% % %     figure % subplot(3,1,1)
% % %     plot(traw,mean(d')), hold on
% % %     title(sprintf('Mean Subj: %d Wndow: %d days', sid, w));
    if plotFlag == 1
        XTicksLabels = datenum(get(gca,'XTick'));
    end

% % %     set(gca,'XTickLabel',datestr(XTicksLabels,2));
% % %     ylabel('Mean Interstroke Time [msec]');

    [xhat medhat sdhat tt] = movestat(mean(d'), t, w);
    tt = tt + traw(1);
% % %     plot(tt,xhat, 'r', 'LineWidth', 3)
% % %     plot(tt,sdhat, 'k', 'LineWidth', 3), hold off

% % %     figure %  subplot(3,1,2)
% % %     plot(traw,scores(:,1))
% % %     title(sprintf('P1 Subj: %d Wndow: %d days', sid, w));
       if plotFlag == 1
         XTicksLabels = datenum(get(gca,'XTick'));
       end

% % %     set(gca,'XTickLabel',datestr(XTicksLabels,2));
% % %     ylabel('P1 [msec]');

% % %     figure %  subplot(3,1,3)
% % %     plot(traw,scores(:,2))
% % %     title(sprintf('P2 Subj: %d Wndow: %d days', sid, w));
      if plotFlag == 1
          XTicksLabels = datenum(get(gca,'XTick'));
      end
% % %     set(gca,'XTickLabel',datestr(XTicksLabels,2));
% % %     ylabel('P2 [msec]');

%     Rpatid is the OADC (Oregon Alzheimer's Disease Center) number of the
participant.
%     Every participant who has a neuropsych done recieves one of these
numbers.
%     There is a link between these two numbers in the subjects_new table.

% run query getSubjectOADC() calls that link and returns
% mysql(SQLQueryOADC)
%
%   subjectID   OADC
%  +----------+ +------+
%   444         11063

      [BRPOutput OADCOutput] = getSubjectOADC(num2str(sid));
%     BRPOutput = current BRP subject ID
%     OADCOutput = current corresponding OADC subject ID

[fttRawData] = csvread('./ftt.csv');

[FttNonDom FttDom] = getFttTimes(fttRawData, OADCOutput);

FttNonDomMS = RawFtt2IntraStrokeAvgMS(FttNonDom);
FttDomMS = RawFtt2IntraStrokeAvgMS(FttDom);

IdAndFttDataMatRaw = [IdAndFttDataMatRaw; BRPOutput OADCOutput FttNonDom
FttDom];

IdAndFttDataMatMS = [IdAndFttDataMatMS; BRPOutput OADCOutput
RawFtt2IntraStrokeAvgMS(FttNonDom) RawFtt2IntraStrokeAvgMS(FttDom)];

    colors = [ [0 0 0]; 0.85*hsv(nkeys-2)];  % Add black, make them darker
    colr = ['r', 'g', 'b' 'c' 'm' 'k'];
    clear legstr
    if plotFlag == 1
        figure
    end

  s4 = ['legList =  userName' num2str(sid)];
  eval(s4);

% To be used to store the max distribution for each letter.
maxOfKeystrokes = [];

        for i=1:(nkeys-1)
            [pdf,xi] = ksdensity(d(:,i));
            if plotFlag == 1
            plot(xi, pdf, 'Color',colors(i,:), 'LineWidth', 2), hold on
            end
    %       Change code to make legend reflect actual keystroke (e.g. "G")
            char1 = legList(i);
            char2 = legList(i + 1);
            legstr{i} = sprintf('%c -> %c', char1, char2);

   %         Find the max of the distribution for each letter.
            [maxNum, indexOfMax] = max(pdf);
            peakKeyStroke = xi(1,indexOfMax);
            maxOfKeystrokes = [maxOfKeystrokes; peakKeyStroke i double(char1)
double(char2)];

        end

    s5 = ['user' userIDstring 'MaxOfPdf = maxOfKeystrokes;'];
    eval(s5);

    if plotFlag == 1
        legend(legstr);
        title(sprintf('PDF Subj %d ', sid));
        xlabel('Interstroke Time [msec]');
    end


%   We'll take the max(xi/pdf) for each keystroke (eg. E -> D, D -> D)
%   distribution, and then we'll put those in a matrix.
%   maxOfKeystrokes
%   e.g
%   e->d  45
%   d->d  657
%   d->i  1111
%   i->e  35
%   THEN ake the ms# from the OADC finger tapping speed (both the dominant
%   and non-dominant hands. Then we take  the ms# from the OADC and subtract
%   the #'s in the e-d matrix from the OADC MS#.
%   Take the ABS value of those #'s, and take the smallest.

[NonDomPeak  IndexOfNonDomPeak] = min(abs(FttNonDomMS  -
maxOfKeystrokes(:,1)));
[DomPeak  IndexOfDomPeak] = min(abs(FttDomMS - maxOfKeystrokes(:,1)));
charKeys = maxOfKeystrokes(IndexOfNonDomPeak,3:4);

% Save the value in a new matrix - one for each user(sid).
UserPeaks = [UserPeaks; sid NonDomPeak DomPeak IndexOfNonDomPeak (nkeys -1)
charKeys];
UserPeaksTest = [UserPeaksTest; sid DomPeak FttDomMS];
UserPeaksTest2 = [UserPeaksTest2; sid DomPeak FttDomMS
maxOfKeystrokes(IndexOfDomPeak,1) numPoints];

% % % % % % % % % % PLOTTING!
      for i=1:(nkeys-1)
            [pdf,xi] = ksdensity(d(:,i));
            if plotFlag == 1
           ttt =  plot(xi, pdf, 'Color',colors(i,:), 'LineWidth', 2), hold on
            plot_lines(FttDomMS)
            end

            char1 = legList(i);
            char2 = legList(i + 1);
            legstr{i} = sprintf('%c -> %c', char1, char2);

  %         Find the max of the distribution for each letter.
            [maxNum, indexOfMax] = max(pdf);
            peakKeyStroke = xi(1,indexOfMax);
            maxOfKeystrokes = [maxOfKeystrokes; peakKeyStroke i double(char1)
double(char2)];
      end
      disp 'plotting 1 user done.'

    sidString = num2str(sid);

%   Save graphs as an image file. e.g. saveas(gcf, sid,'jpg');
    s6 = ['saveas(gcf,' ' ''' sidString   ''' '     ',' ' ''jpg'')   '  ];
    eval(s6);

    if (sid == subjs(1))
        close
    end
    close
% % % % % % % % % % END PLOTTING
end

% Writes IdAndFttDataMat to IdAndFttDataMat.csv
dlmwrite('./IdAndFttDataMatRaw.csv',  IdAndFttDataMatRaw, 'precision',
'%15.7f');
dlmwrite('./IdAndFttDataMatMS.csv',  IdAndFttDataMatMS, 'precision',
'%15.7f');
dlmwrite('./UserPeaks.csv',  UserPeaks, 'precision', '%15.7f');
dlmwrite('./UserPeaksTest.csv',  UserPeaksTest, 'precision', '%15.7f');
dlmwrite('./UserPeaksTest2.csv',  UserPeaksTest2, 'precision', '%15.7f');
toc
```

movestat.m

```
function  [xhat medhat sdhat tsample] = movestat(x, t, w)
% x is input irregulrly sampled at points t;
% w is the window width,,  , type_str
% Determine sample points
tmax = t(end);
tt = unique(round(t));
stepsz = w;

% Compute estimates of the central tendencies
wn = 0;
for ti = 1:stepsz:tmax
    % comute sum
    wn = wn + 1;
    tsample(wn) = ti;
    idx = find((t>(ti-w)) & (t<=ti));
    npt(wn) = length(idx);
    if npt(wn) > 1
        xhat(wn) = mean(x(idx));
        sdhat(wn) = std(x(idx));
        medhat(wn) = median(x(idx));

    else
        xhat(wn) = NaN;
        sdhat(wn) =  NaN;
        medhat(wn) = NaN;
    end
end
```

getSubjectOADC

```
function [BRPOutput OADCOutput] = getSubjectOADC(subjectID)
%getSubjectOADC Use BRP ID get OADC ID from BRP DB
%   This function sends the BRP subject ID number to the function call
(getSubjectOADC)in
%   the BRP database and the OADC number is returned.

% A function was added the BRP called 'getSubjectOADC'.
% eg: SELECT subjectId, getSubjectOADC(subjectId) as OADC FROM KCLogin;

% Returns the OADC ID number
% SELECT  DISTINCT subjectId, getSubjectOADC(444) as OADC FROM KCLogin where
subjectId = 444;

SQLQueryOADC = ['SELECT DISTINCT subjectID , getSubjectOADC(' subjectID ') as
OADC FROM KCLogin WHERE subjectId = ' subjectID ';'];
[DBOutputSID DBOutputOADC] = mysql(SQLQueryOADC);

BRPOutput = DBOutputSID;

OADCOutput2WorkingStep = mat2str(cell2mat(DBOutputOADC));

OADCOutput = str2num(OADCOutput2WorkingStep(2:end-1));
```

getFttTimes

```
function [FttNonDom FttDom] = getFttTimes(fttRawDataMatrix, OADCOutput)
%getSubjectOADC Use BRP ID get OADC ID from BRP DB
%   This function sends the BRP subject ID number to the function call
(getSubjectOADC)in
%   the BRP database. This function returns the ??

for i=1:length(fttRawDataMatrix)
    if fttRawDataMatrix(i,1) == OADCOutput
      FttNonDom = fttRawDataMatrix(i,2);
      FttDom = fttRawDataMatrix(i,3);
    end
end
```

RawFtt2IntraStrokeAvgMS

```
function [ MSTime ] = RawFtt2IntraStrokeAvgMS(RawTime)
%RAWFTT2INTRASTROKEAVGMS Changes the Raw FTT times (number of taps per
%10 seconds) and returns avg time between fingertaps in milliseconds.

MSTime = ( 1 / ((RawTime * 6) / (60 * 1000)));
```

plot_lines.m

```matlab
function h = plot_lines(xin,mnmx,lntype);

%plot_lines(xin)    Plots vertical lines at points defined on the x-axis
%   using red solid lines and the current y-axis limits using a single line
%   handle
%plot_lines(xin,[a b])  Plots the lines from a to b
%
%plot_lines(xin,[],lntype)  Plots the lines using the current y-axis limits
%using the line type deined by lntype
%
%h = plot_lines(xin,[a b],lntype)  Returns the handle to the vertcal lines
%
%Ex.)
%figure(1);clf;
%plot(randn(1,1000));
%x = [10 170 300 450 600 800 990];
%hold on;
%plot_lines(x,[],'b:');

%Written by S. Wegerich, SmartSignal, Corp. 09/03


if((nargin<2)|isempty(mnmx))
    mnmx = get(gca,'YLim');
end

if((nargin<3)|isempty(lntype))
    lntype = '-r';
end

xin = xin(:)';
L = length(xin);
x = reshape([xin;xin;ones(1,L)*nan;],L*3,1);
x = x(1:end-1);
x = x(:);
mnmx = mnmx(:)';

y = repmat([mnmx nan],1,L);
y = y(1:end-1);

h = plot(x,y,lntype);
```

R: The R Project for Statistical Computing - graphing scripts

```
USERNAME
data3 <- read.table('UserPeaksTest2edited.csv', sep=",", header=FALSE)

ftt <- data3$V3
ks <- data3$V4

plot(ftt, ks,
xlab="Avg Finger Tapping Test speed - dominant hand",
ylab="Average typing speed for repeated letter",
main="Subjects with repeating letters in username.
E.g. (Terry).
Time in milliseconds, 15 users",
)

res <- lm(ks~ftt)
summary(res)
abline(res)
========
UserPeaksTest2edited.csv

PASSWORD
data3 <- read.table('UserPeaksTest2edited.csv', sep=",", header=FALSE)

ftt <- data3$V3
ks <- data3$V4

plot(ftt, ks,
xlab="Avg Finger Tapping Test speed - dominant hand",
ylab="Average typing speed for repeated number",
main="Subjects with repeated numbers in password.
E.g. (1337).
Time in milliseconds, 24 subjects",
)

res <- lm(ks~ftt)
summary(res)
abline(res)
```