

**INDIANA UNIVERSITY'S
ADVANCED CYBERINFRASTRUCTURE**

THE LEAST YOU NEED TO KNOW

Last revised: 1 August 2009

Preface

For most researchers at Indiana University and the US as a whole, information technology is a tool – a means, not an end. This point of view is reflected in the term cyberinfrastructure, which IU defines as high performance computers, massive data storage systems, data resources, advanced instruments, sensor networks, and *people* all linked together by advanced software and high performance networks to improve research productivity and enable breakthroughs not otherwise possible. Ideally, a researcher should have to think about computing power and data storage in no more detail than is given to thinking about the generation of electrical power that feeds the outlets in a lab. While computer scientists and IT researchers at Indiana University and elsewhere are striving toward this goal, we're not there yet.

The purpose of this document is to introduce researchers to Indiana University's cyberinfrastructure – to clarify what these facilities make possible, to discuss how to use them and the professional staff available to work with you. It may seem incongruous to find a document as long as this subtitled "The least you need to know" but the resources described here are complex and varied, among the most advanced in the world.

The intended audience is anyone unfamiliar with IU's cyberinfrastructure. Many readers will find sections (for example, introducing UNIX) they are familiar with and want to skip. In some cases there is information that does not need to be learned, but will help with a one-time process: logging into a particular computer for the first time, for example, involves steps that may be followed keystroke-by-keystroke and then forgotten.

This document is intended to be a starting point, not a comprehensive guide. As such it hopes to get you off to a good start, and then point you in the direction of online resources and UITS consulting staff. It is provided for the convenience of researchers, to permit anyone at all familiar with computers and the Internet to get a general overview of IU's cyberinfrastructure and what it offers. References to online information are given, but it is our intention that you can print this and carry it with you, to read without network access.

1	Computing at IU	1
1.1	Introduction	1
1.2	Computing support at IU	2
1.3	Slashtmp	3
1.4	Grid computing	3
1.5	Videoconferencing	3
1.6	Typographic conventions	3
1.7	Terminology: bits, bytes, and FLOPS	4
2	Research Technologies services	5
2.1	Why use supercomputers?	5
2.1.1	Single-processor performance	6
2.1.2	Trivially parallel processing	6
2.1.3	Not-so-trivial Parallel programming	6
2.2	Data storage and management	7
2.3	Databases	7
2.4	Consulting and support	8
3	Getting started with IU's research computers	9
3.1	Requesting research accounts at IU	9
3.2	Connecting to research systems at IU	9
3.2.1	Logging in from Windows	10
3.2.2	Logging in from Macintosh or Linux	10
3.3	First time login	10
3.4	Second login	11
3.5	Logging out	11
3.6	Basic interaction with UNIX operating systems	11
3.7	Editing files	12
3.7.1	A common source of confusion	13
3.8	Files and directories	13
3.9	Stumbling blocks	14
3.9.1	Common error messages	14
3.9.2	Commands to handle with care	15
3.9.3	Files to handle with care	15
3.9.4	What to do when confusing things happen	15
3.10	Printing files	16
3.11	Transferring files to or from research systems	16
3.11.1	From Windows	16
3.11.2	From Mac OS	17
3.11.3	From UNIX	18
3.12	Where should you keep your data?	18
4	Running programs	19
4.1	Compiling FORTRAN, C, C++ programs on Quarry	19
4.2	Batch processing	19
4.2.1	Serialjob	19

4.2.2	Submission scripts for the Portable Batch System.....	20
5	Big Red.....	21
5.1	First time log in.....	21
5.2	Compiling.....	22
5.3	Loadleveler.....	22
5.4	Softenv.....	24
6	Commercial programs.....	25
6.1	SAS.....	25
6.1.1	Running SAS Jobs.....	26
6.1.2	Interactive and background jobs.....	26
6.1.3	Batch jobs.....	26
6.2	Matlab jobs.....	27
6.2.1	Interactive and background processes.....	27
6.2.2	Batch processes.....	28
7	Archival data storage.....	29
7.1	Requesting accounts.....	30
7.2	Accessing research data storage systems.....	30
7.2.1	Accessing the Massive Data Storage System with Secure FTP.....	30
7.2.2	Mounting the Research File System on your desktop system.....	31
7.2.3	Hierarchical Storage Interface.....	32
8	High speed temporary storage.....	33
9	Using IU's visualization facilities.....	34
9.1	Supported visualization platforms.....	34
9.2	Visualization services.....	34
9.3	Visualization software.....	35
9.4	Visualization facilities.....	35
10	Contact information.....	36
	A final note.....	37
	Acknowledgments.....	37

Changes from January 2009:

- The Libra Cluster has retired.
- Web pages moved from rtinfo to ptl.
- Minor editorial changes, including this list of changes.

Last revised: 1 August 2009

INDIANA UNIVERSITY'S

ADVANCED CYBERINFRASTRUCTURE

1 Computing at IU

1.1 Introduction

Indiana University has a cyberinfrastructure that is among the best at any university in the world, including supercomputers, data storage systems, visualization environments, and access to high performance research networks. These facilities have been put in place in order to aid you, the researcher – to make possible new calculations, analyses and visualizations of massive amounts of data, to assist researchers to achieve breakthroughs in their scholarship. The purpose of this document is to enable researchers to as easily as possible learn about, get accounts on, and get started using IU's cyberinfrastructure. The subtitle of this document reflects the goal of getting you started using these facilities with just the minimal necessary information – nothing extraneous.

To see why you might care about these systems, ask yourself:

- What are you doing with computers now,
that you would like to do faster, in more detail, or on a larger scale?
- What kind of research questions would you tackle,
if computing resources and expertise were not a limiting factor?

Read this document to get a better idea as to the kinds of computing made possible through use of UITS' advanced IT facilities, and whether you can use any of it. Most of the services that the Research Technologies Division of University Information Technology Services (UITS) offers are baseline services, that is, UITS provides computing resources at IU in a way much like the Library provides books and journals. Baseline services are almost always offered without any direct charge to the user, as long as the purpose is research or education. UITS provides hardware, software, networks, *and support*, to enhance research and instruction.

Please note that this document is not intended as a comprehensive guide to IU's information technology environment – it focuses only on *research* technologies. General purpose computing (for example, e-mail) is important, but a document describing *all* of UITS' services would be *very* long and possibly somewhat boring. Research computing systems are not intended for e-mail, and do not accept it. This document does *not* discuss email issues (although this first section does include some pointers to where to get help with them, along with other background information). A description of services available through UITS is available online at:

<http://uits.iu.edu/>

Research Technologies is part of the Pervasive Technologies Institute. A description of services available through Research Technologies is available at:

<http://pti.iu.edu/rt/>

For information not contained in those Web pages or this document, e-mail sent to researchtechnologies@iu.edu is always welcome.

We gratefully acknowledge the support of the National Science Foundation, the Lilly Endowment, Inc., the National Institutes of Health, IBM, Inc., StorageTek, and Sun Microsystems; as well as support from the State of Indiana and Indiana University itself. Without the generosity of all of these organizations, we would not be able to offer the systems and services discussed in this document. The support of the Lilly Endowment, Inc. for the Pervasive Technology Labs, the Indiana Genomics Initiative, and METACyt have been particularly important in advancing IU's strategic goals in advanced information technology and life sciences. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of those organizations.

1.2 Computing support at IU

Support for computing at IU is provided through a tiered system, starting with the Local Support Providers in departments, who set up Windows and Macintosh workstations and connect them to the campus network. UITs provides more specialized resources, to the university as a whole.

Basic computing information is available from the UITs Support Center and Knowledge Base. The UITs **Support Center** is available at 812-855-6789 or 317-274-HELP, 24 hours a day, 7 days a week (or send mail to ithelp@iu.edu). They provide excellent support for Windows, Macintosh, Linux and Unix computing, but not for research systems – at least not in depth (that's what *this* document is mostly about).

The IU **Knowledge Base** (KB) is an award-winning source of information, online at:

<http://kb.iu.edu/>

It's much more than Frequently Asked Questions. The KB is our primary means for delivering documentation. It contains over 12,000 answers to questions about information technology. You can find answers to their questions through several different interfaces. A menu-based interface sorts common questions by topic (easiest for beginners), full-text search returns documents matching your search terms (most efficient for experienced users), and a glossary defines important terms.

We assume that you have a personal computer, from which to access the research computing, data storage and visualization systems (this is discussed in more detail below). Beginners sometimes ask about physical access to IU's computing facilities. You do not need to go to any "Computing Center." ***Access to research computing systems is through the network, using your personal computer, or cell phone, etc***, and there is no need to ever visit, or even know the location of, a supercomputer or storage disk farm. It is not likely that you will notice any difference between using a system in Bloomington and using one in Indianapolis. But you are welcome to visit!

This document assumes that you already have your basic computer and email account, referred to at IU as your **Network ID**, or just NetID. The KB document "How do I get my first computing accounts at IU?" provides advice on this topic:

<http://kb.iu.edu/data/achn.html>

1.3 *Slashtmp*

Slashtmp is a service that allows IU grad students, faculty, and staff to share files via a web interface. It works for files that are too large for email. Detailed instructions are available at:

<https://www.slashtmp.iu.edu/>

1.4 *Grid computing*

A “grid” is a way to combine supercomputers in multiple locations, managed by multiple organizations. IU is part of national computing projects named the **TeraGrid** and the **Open Science Grid**, multi-year efforts to build and deploy the world's largest and most comprehensive distributed infrastructure for unclassified scientific research. These grids provide many teraflops of computing power, facilities capable of managing and storing several petabytes of data, high-resolution visualization environments, scientific instruments, large datasets, and computing portals and toolkits. Their components are tightly integrated through high capacity networks. Grid computing is beyond the scope of this document; a separate document is planned. The Research Technologies division provides support for TeraGrid users, as discussed below.

1.5 *Videoconferencing*

Indiana University offers several videoconferencing services, including administrative group videoconferencing, desktop videoconferencing, and streaming media. UITS Digital Media Network Services (DMNS) – not a part of Research Technologies – has evaluated a variety of products and strongly recommends Polycom ViaVideo for university-based desktop video conferencing. More information is available on the web page at:

<http://www.indiana.edu/~video/>

1.6 *Typographic conventions*

We will present examples as follows:

Text that appears on the screen is in Courier font.

Text that you are to type verbatim is boldfaced.

Text that may change, such as a name, is italicized.

Combination keystrokes are joined with a slash. For example, **CTRL/z** means hold down the CTRL key while you press the z key (sometime “^Z” is used to indicate this).

When you are instructed to *enter* some text, do so and then press RETURN or ENTER (depending on your keyboard). When you are instructed to *press* a key, press that key alone, and do not press RETURN.

1.7 Terminology: bits, bytes, and FLOPS

Computing has over time developed its own private dialect. You do not need to memorize these terms, to understand this document, but a comparison of data sizes and performance measurements is of some interest:

A **Bit** is the fundamental (and smallest) unit of data representation within a computer. A bit is always either a 0 or a 1.

A **byte** is the smallest unit of data representation usually used by a computer. A byte consists of 8 bits and represents roughly one alphabetical character, such as an a or b, or a digit (like 2 or 7) or punctuation mark, such as a comma. Eight bits gives 2 to the 8th, or 256, characters in the standard English computer alphabet.

A **Megabyte**, abbreviated **MB**, is a million bytes, or that same number of characters. There are about 700 Megabytes of data on a typical CD.

A **Gigabyte (GB)** is a billion (10^9) characters.

A **Terabyte (TB)** is a trillion (10^{12}) characters. All of the printed text in the IUPUI library is about 1TB.

A **Petabyte (PB)** is a quadrillion (10^{15}) characters. The printed text in all U.S. academic libraries is about 2PB.

FLOPS – Floating Point Operations Per Second. A floating-point operation is multiplying two decimal numbers together, like 2.78 times 3.14.

Gigaflops – one billion (10^9) FLOPS. If a computer performed one floating-point operation per clock tick, then the FLOPS count would always be the same as the processor chip's clock rate. In practice this isn't so. The theoretical peak FLOPS rate can be faster for processors that perform more than one mathematical operation per clock tick. In any case, computers never achieve their theoretical peak FLOPS count.

Teraflops – one trillion (10^{12}) FLOPS. Indiana University's Big Red supercomputer has a theoretical peak performance of 30 Teraflops.

2 Research Technologies services

As with UITS services generally, advanced IT services are available without charge for research and instruction. The Research Technologies division's services are focused on a few main areas:

Supercomputing – By using IU's supercomputing facilities, researchers can drastically accelerate their simulations and analyses, performing massive new computations that are not otherwise practical.

IU has two supercomputers: Big Red and Quarry. They are built from many processors, which are organized into "nodes." More information on these systems is included in later sections of this document, and online. The platform or platforms you choose to use will be a matter of your programs' requirements in terms of operating system, software packages, computing power and available memory, as well as your own personal preferences. You can ask for advice.

Data Storage and Management – IU's data storage facilities make possible the secure and reliable storage of massive amounts of data. Anyone who uses more data than fits conveniently on a single CD can benefit from using these systems. And anyone who has ever said, "I'd like to pursue a research idea, but can't store that much data" will most likely find that storage is no longer a limiting factor. In addition, UITS supports use of research databases, and publishing data via the Web.

Consulting – Rather than one researcher at a time climbing up the learning curve, specialists can assist, benefiting the entire community – this is the fundamental model underlying the university. Research Technologies provides consulting and support for:

- scientific visualization, virtual reality, and high-end computer graphics
- statistical, mathematical or bioinformatics software
- data storage and database design
- programming supercomputers.

This is *not* an exhaustive list.

2.1 Why use supercomputers?

Supercomputers (also called *high performance* computers) make it possible to run software applications more quickly and to perform computations that would otherwise take unreasonably long. For example, Research Technologies staff worked with one School of Medicine researcher who uses a program that typically takes a week to analyze a single data set on the computer in his lab. The program now finishes in an hour.

There are three main ways that supercomputers make programs run faster: better single-processor performance, trivially parallel processing, and parallel programming.

2.1.1 Single-processor performance

Programs will often run faster on a supercomputer than on other computers for three reasons:

- Due to the chip *design*, a combination of speed, caching and other optimizations in the hardware, as well as optimizations of software, numerically intensive tasks often run faster on supercomputers than on personal computers.
- The supercomputer has more *memory* than most computers, so it can hold more data in memory at a given time. Your personal computer might have to write pieces of intermediate calculations out to its hard disk due to limited memory.
- A supercomputer may have better hard *disk performance* than most desktop systems. There are a number of features engineered into IU's supercomputers that make transferring data to and from the hard disk very fast.

All of these factors make supercomputers faster than microcomputers or departmental servers for many purposes – but unless your workstation or departmental server runs the UNIX operating system, the environment provided by supercomputers will differ from what you are used to. This document is intended to help you over this "threshold," by familiarizing you with the UNIX environment.

Also, many commercial programs that run on Microsoft or Apple operating systems (such as SAS or Matlab) also run on UNIX in exactly the same way.

Finally, if you have programs that you or a colleague has written for other UNIX systems, modifying this program to run on IU's supercomputers ("porting" the code) is typically very easy (and something Research Technologies staff will gladly help you do). Programs written for Microsoft DOS or Windows, or Apple Macintosh, can usually be ported, as well. The simplest method of speeding up your computing may be to take a program that you are currently running on a personal or a departmental computer, and with the help of UITs staff, get it to run on one of IU's supercomputers.

2.1.2 Trivially parallel processing

Suppose you have a program that runs on a single processor, and you have many different data sets to analyze. Thanks to the many processors of IU's supercomputers, you can actually run the same program dozens of times simultaneously, each time analyzing a different data set. This could be done by repeatedly submitting jobs with varying input or parameters, but the process can usually be automated using simple programs called "shell scripts." This is the most common and most easily mastered form of "parallel computing."

Trivially parallel processing is a computer science term – "perfectly parallel processing" might be more descriptive. This is a very straightforward way to achieve tremendous speedups in your data analysis. Research Technologies staff will help you implement the scripts necessary to do this sort of parallel processing, usually in a matter of days.

2.1.3 Not-so-trivial Parallel programming

It is not always a simple matter to organize the work a program does so that you can cut the time required in half, by doubling the number of processors working on it. In fact, that is the theoretical limit of what can be done, and this limit is rarely achieved.

This is complicated work requiring expertise, and it can take weeks or months to convert from running on a single processor to running efficiently on multiple processors simultaneously. Parallel programming is not simple, but it may be the key to speeding up your applications, and Research Technologies staff members are available work with IU researchers to parallelize your programs, so that you can finish your work more quickly.

2.2 Data storage and management

A researcher today may deal with all sorts of computer data – data from field research, from instruments, from surveys, data generated by computer simulations, etc. A lot of data ultimately ends up on CDs or the hard disk of a personal computer or on a departmental server. The time and discipline, and the cost of the hardware required to backup this data may be formidable, which often means that, in practice, there are no backups available. Consider the following situations:

- You are unable to take on a project because you do not have enough storage;
- You have an instrument in your lab which produces data at a rate or a volume that you cannot handle;
- You want to access your data securely and transparently, from anywhere;
- You need a place to store your data while out of the country;
- You want truly reliable backups.

If any of these apply, Research Technologies can help. We won't back up your data for you, but we'll give you a safe and convenient place to put it.

Many people are familiar with storing data by writing it to a CD or DVD, which hold about 700 MB or 4.7GB of data, respectively. Some instruments are set to write the results of a particular analysis on a disk, which is then handed to the researcher. Data retrieval can be agonizingly slow. UITS provides specialized systems capable of capturing data from modern digital instruments and accessing it again at GB/sec rates. The storage capacity available is world-class. More details are given in the "Archival data storage" section, below.

2.3 Databases

The Research Database Complex (RDC) is configured to support both Oracle- and MySQL-based databases. If you have data that you would like to be able to store, query, and retrieve easily, UITS can set up a database on these systems. The host, disk space, database management software, backups and patches are all provided. These databases can be set up so that only those people you specify can access them. By using these central systems, you can have the advantage of extremely good database performance, database administration provided by UITS, and professionally-managed backup.

We are able to keep local "mirror" copies of public databases. If you make extensive use of a public database and would like to have a locally accessible copy, we can provide that, if the owners permit it. We can also provide access to licensed data: for example, the Centralized Life Sciences Data (CLSD) service provides access to a wide range of public genomics data (some

mirrored locally and some accessed externally). CLSD is ideal for researchers who want to write their own programs to access large amounts of regularly updated public genomics data.

2.4 Consulting and support

The Research Technologies division can be roughly split into the groups running the hardware, and the groups supporting researchers' applications (software), for whom consulting is paramount.

- The staff of the **UITS Advanced Visualization Laboratory (AVL)** is available to help you with the latest visualization techniques and technologies. Visualization allows researchers to analyze their data in whole and in detail, and to rapidly spot trends, recognize relationships, and identify anomalies. Specific services and facilities offered are detailed later in this document, in the section "Using IU's visualization facilities."
- **Bioinformatics Support** provides help in biological computing, particularly in the areas of genomics, cell biology, and molecular biology.
- **The Biomedical Applications** group provides consulting and custom development to help biomedical researchers access and manage their data.
- The **Center for Statistical and Mathematical Computing** provides help with statistical and mathematical software (such as SAS, Mathematica or Matlab), on personal computers as well as supercomputers. If you want this software for your department or personal computer, they make very favorable licensing arrangements available.
- The **Digital Library Program**, a joint effort of the IU Libraries and UITS, provides consulting and other services related to digital library development.
- **The High Performance Applications** group provides programming support, and can answer questions you may have about how to gain the advantages supercomputers can offer. They will help you to migrate, optimize, and parallelize your code, and they manage licenses for programmers' tools (such as compilers, libraries, debuggers and performance analyzers). This group will also help you use the resources of the TeraGrid.
- Research Technologies **Core Services** provides enterprise Linux licensing [RedHat, SUSE], plus public access to Free/Open Source Software, a source code repository/ version control system, and a Condor-based rendering service.

These groups are interested in long-term projects that offer opportunities to acquire expertise that can be made broadly available to the university community. They include many Ph.D.s, and frequently participate in externally funded projects.

3 Getting started with IU's research computers

IU has two supercomputers: Big Red and Quarry. We will start with Quarry, the most widely accessible (including undergraduates), but we want to stress that both systems appear to be mostly the same – the compilers and job management systems vary, but there is very little difference in interacting with them. IU's research computers all run variants of the LINUX or UNIX operating system.

Experienced users will find most of this chapter routine, but should look at the final section (Where should you keep your data?).

3.1 Requesting research accounts at IU

To obtain an account on Quarry, Big Red, or the RDC, use the Account Management Service at:

<https://itaccounts.iu.edu/>

- Select Manage my IU computing accounts.
- If you are prompted to log into the Central Authentication Service, enter your Network ID username and passphrase. If not, skip to the next step.
- Click “create more accounts.”
- Use the radio buttons to select the accounts you want to create, then click Create Account.

As mentioned earlier, these services are available without charge, for research and education. You need only:

- Acknowledge use of IU's cyberinfrastructure in your publications.
- Respond to occasional requests for citations, so that IU can measure the impact its cyberinfrastructure is having on scholarly activity.

3.2 Connecting to research systems at IU

A goal of the Research Technologies division of UITS is to provide robust, reliable services to the IU community. A critical part of making any computer system reliable is keeping it secure. In order to protect UITS research systems, all sessions must be encrypted, by using a protocol called **Secure Shell**. The program for connecting between computers, or "terminal emulation," using this protocol is often called SSH, as well. If you do not have SSH installed on your personal workstation, it's available from a service provided by UITS named **IUware**, at:

<http://iuware.iu.edu/>

The *most common source of confusion* for new users of UNIX systems is between programs running on your personal computer, and programs running on another computer, especially when both are making things happen and you can interact with either! This affects cut and paste editing operations, for example, and is discussed more below (in the subsections "What to do when confusing things happen?" and "Editing files").

3.2.1 Logging in from Windows

To log into a research system from a Windows workstation, select the SSH Secure Shell Client icon on your computer. When the program opens, enter the host name **quarry.uits.indiana.edu** and your *username*, leave the port at its default value (22), and select the authentication method "Password."

The *first* time you connect to this system, a window will appear asking if you want to save the host key; select yes. Another window will pop up, asking for your password; enter it. Finally, you will be offered the opportunity to save a new "profile" – enter the word **Quarry**. After the first time, you can select the "Profiles" menu, select Quarry, and go straight to Password.

3.2.2 Logging in from Macintosh or Linux

If your personal workstation has the Linux operating system or Mac OS X, log into a research system by opening a terminal [window]; at the prompt enter:

```
ssh quarry.uits.indiana.edu
```

You will be asked for your password. Enter it [you will see nothing, or ***'s], and you will be greeted with the Message of the Day (*read it, please!*). [Or possibly an message, for example if you made an error – in which case, try again]. When you are logged in, you will see the "UNIX prompt" – except for the very first time you log in, which is discussed next.

3.3 First time login

When you use a computer that runs the UNIX operating system you interact with a piece of software known as *the shell*, the public face of UNIX. The shell is text oriented – it was designed before the "mouse" was available. (Note: The "UNIX shell" that you use to interact with the operating system is different from the "Secure Shell" that you use to connect your desktop computer to the UNIX system).

Actually there is more than one possible shell program available. The very first time you log in to a UITS supercomputer, a program named **firstshell** is executed, which asks you to choose your default shell. Unless you are an experienced UNIX user and have a specific reason to do otherwise, we strongly recommend that you select the "bash shell" on Quarry. The first time you log in, you will be presented with the following greeting:

```
This program is run the very first time you log in
to Quarry to allow you to select your login shell.
If you are uncertain which shell to select, choose
bash (Bourne-again shell).
```

- 1) bash
- 2) tcsh
- 3) ksh
- 4) zsh
- 5) quit

```
Select 1-5:
```

Enter: **1**. You will see:

```
Changing shell for username.
```

```
Your shell has been changed to bash
This will take effect on all nodes within 60 minutes
Select 1-5:
```

Type **5** and press enter to exit the system. You will receive the message:

```
Connection to quarry.uits.indiana.edu closed.
```

Once you select your preferred login shell it will be automatically set on the node you're currently on. It might take up to 60 minutes to be propagated to all nodes.

Each time you login thereafter, you will see the Message of the Day, which displays current announcements. Please read it.

3.4 Second login

To verify your shell selection, log back into Quarry again, and at the prompt enter:

```
echo $SHELL
```

The response should be:

```
/bin/bash
```

3.5 Logging out

To end your UNIX session, enter **logout**.

If the system ever responds with `There are suspended jobs`, simply type the command `logout` again. Be sure always to log out when you are done using a system. Never leave an active session unattended!

3.6 Basic interaction with UNIX operating systems

The UNIX operating system is a powerful environment designed by and for computer programmers. Beginners and casual users often find the command-line interface a hurdle, and the jargon-filled help system frustrating, but remember, many users spend little time at the UNIX command line, working instead within applications like SAS. This document will show you how to work with the UNIX command line, and get into the applications that you need.

When you log in to an interactive session, the UNIX system runs a "shell" program that puts a prompt on your terminal screen and processes your commands. Prompts may vary (some common ones are \$, %, and >). In this document the system prompt is represented as \$.

Programs are ran by entering their name at the command prompt, as for example:

```
$ hostname          [The shell program supplied the $, you enter "hostname"]
b003                [This is the system's response, it may vary]
```

The \$ character is the prompt, that indicates that the shell is ready for a command; we will usually not include it in our examples, after this section. The response shows that I am connected to the "003" node of Quarry (you may get a different node – a node is one of the parts of the system, and is of no particular interest in itself).

UNIX is case sensitive, meaning that when you type a command, you must use uppercase or lowercase letters as they appear in this document. Most UNIX commands are lowercase only.

There is an online manual for particular UNIX commands: at the UNIX prompt, type **man** and the name of the command, and then press the ENTER or RETURN key. For example, to get help with the man command itself, you would enter:

```
$ man man
```

In this example you would see the manual page for the man command; the italics indicate that you can replace *man* with another manual subject, for example **man date**, to learn about the date command. **When the man program is running, you press the spacebar to see additional pages, and type q to exit the manual.**

This is an important point: once you start the man program, you're in a different environment than when you were at the shell prompt. Since UNIX was designed before graphical user interfaces were developed, you have to keep track of these things as you proceed.

3.7 Editing files

We recommend that you use an editing program called **Nano** on Quarry, to create a new file or change the contents of an existing file. To edit a file named *note-to-myself.txt* with Nano, at the prompt enter:

```
nano note-to-myself.txt
```

Nano either opens the file *note-to-myself.txt* (if it exists), or creates a new, empty file with that name. Nano fills your screen with the contents of the file *note-to-myself.txt* and displays a command summary at the bottom of the screen (this is considered user-friendly, in some quarters). For example, the first entry in the command summary reads:

```
^G    Get Help
```

This means that to get help, hold down the CTRL key while you press **g**.

To add text to your file, simply begin typing. Use the arrow keys to move the cursor. Here are some basic Nano commands:

CTRL/x	exit Nano (you'll get a chance to save your file)
CTRL/o	save (output) your file
CTRL/v	move forward one screen
CTRL/y	move back one screen
CTRL/r	at the cursor insert another file's contents
CTRL/k	cut the cursor's line of text
CTRL/u	paste the previous cut
CTRL/w	search the file for a text string

3.7.1 A common source of confusion

Your terminal program also offers copy-and-paste functionality. That's on your desktop system, not the UNIX system! Keeping this straight will avoid much frustration.

In particular: the cut and paste commands for Nano store the text in memory on Quarry. Your terminal emulator (SSH) also can cut and paste, but has the data on your workstation. It very likely uses *different* keystroke combinations to do it [look for a drop-down "Edit" menu]. You can highlight text anywhere in the terminal window and copy (but not cut) it with your terminal program; if you then paste it, using the terminal program's keystroke combination, the terminal will send that text to the UNIX program, *at the cursor's current location of that program*, which may well be quite unaware of your mouse.

There are graphical user interfaces for UNIX, but they are beyond the scope of this document. Even so, interaction with a supercomputer is generally text-based, in a terminal window.

3.8 Files and directories

Once logged in, you will be located in your home directory. A UNIX directory is the same as a Windows or Macintosh folder. To find out where you are, *i.e.*, the path to your current working directory, enter:

```
pwd
```

The response will be of the form `/N/u/username/Quarry`.

The shell also refers to the current directory as "." and its parent directory as "..". To move to the *parent* directory of the *current* directory, enter:

```
cd ..
```

Use "~" to refer to your home directory; `~statmath` refers to statmath's home directory. To go to statmath's home directory, enter:

```
cd ~statmath
```

To return directly to your home directory from wherever you are, enter:

```
cd ~
```

To create a new directory, use the **mkdir** command. For example, to create a directory named NewCode, you would enter at the prompt:

```
mkdir NewCode
```

To enter a subdirectory (*i.e.*, make it your working directory), use the **cd** command. For example, to make the newly created NewCode directory your current working directory, enter at the prompt:

```
cd NewCode
```

Filenames may contain letters, numbers, underscores (`_`), dashes (`-`) and dots (`.`) – but no spaces!

Unlike Windows, UNIX does not associate any particular application with a file or filename suffix (like `.doc`). You may however find it convenient to use extensions to keep track of your file types (*e.g.*, you could end all your SAS command files with `.sas`).

To see a list of your files, at the prompt enter:

```
ls
```

You will see a listing of your files and directories, similar to this:

```
Mail                myjob.sas           note-to-myself.txt
```

To view the contents of a file named *note-to-myself.txt*, type:

```
cat note-to-myself.txt
```

Warning: you only want to do this with text files. To do so with a “binary” file could mess up your terminal. The next section discusses ways to recover.

To view the contents of that file, one screen at a time, type:

```
more note-to-myself.txt
```

The first screenful of the file you named will be displayed. To go to the next screen, press the space bar. To go back one page, press **b**. To quit viewing the file, press **q** (enter not required).

To rename a file, use the **mv** (move) command. For example, if you mistyped a filename and called it *ob.sas* when you meant *job.sas*, you can correct this error by entering at the prompt:

```
mv ob.sas job.sas
```

The file is renamed *job.sas*, and there will no longer be a file named *ob.sas*.

To make a copy of a file, use the **cp** command. For example, if you have a file *job.sas*, and you want to make a copy called *job2.sas*, enter at the prompt:

```
cp job.sas job2.sas
```

You now have two files with identical content but different filenames, *job.sas* and *job2.sas*. This is a good way to produce a template, ready for modification.

To delete a file, use the **rm** (remove) command. For example, to remove the file *myshellscript*, you should enter at the prompt:

```
rm -i myshellscript
```

The system will inquire: *remove myshellscript?* If you're sure you want to remove it, press **y**, or if you change your mind and want to keep it, press **n**. The “-i” is an option; the man page for each command given above describes its options.

3.9 Stumbling blocks

3.9.1 Common error messages

UNIX error messages can be frustrating. For example, the command to list the files in your current working directory is “ls”. Here's an example of what can go wrong:

```
$ LS
LS: Command not found.
```

Remember: UNIX is case sensitive. If you have your caps lock set and want to list the contents of your working directory, you must turn off capitalization. Here's another example:

```
$ splus
```

splus: Command not found.

In this example, `Command not found` indicates that "splus" is not recognized; "**Splus**" is the correct spelling.

Error messages often occur because you need to supply one or more parameters. For example:

```
$ mv
Usage: mv [-i | -f] [--] src target
       or: mv [-i | -f] [--] src1 ... srcN directory
```

In this example, when using the "move" command (which renames a file), you must give the old (`src`) and new (`target`) names of the file; the stuff in square brackets is optional (the manual explains them, enter **man mv** to see details).

3.9.2 *Commands to handle with care*

Certain UNIX commands can be confusing for beginners. If someone shows you a new command, be sure to ask if it might be dangerous to use. In particular, you should not experiment on your own with the commands `chmod`, `emacs`, `vi`, or `passwd`.

The UNIX command to remove (delete) a file, **rm**, is final; there is no wastebasket from which to recover the former file. Likewise, if you copy one file over another (use a name that already exists), it's irrevocable. UNIX provides mechanisms for coping with these foibles: for example, `mv`, `rm` and `cp` have an **option to inquire** whether you really want to clobber a file:

```
$ mv -i SomeSillyFile ReallyImportantFile
overwrite ReallyImportantFile?
```

You then can answer **y** (yes) or **n** (no).

3.9.3 *Files to handle with care*

There are configuration files in your home directory, which establish your working environment. These files have names that start with a period, and they are called "hidden" because they are not listed by "ls." They are beyond the scope of this document.

3.9.4 *What to do when confusing things happen*

You can check the Knowledge Base, or call the Help Desk (5-6789 at IUB, 4-HELP at IUPUI).

If the DELETE and BACKSPACE keys don't seem to work correctly, try using **CTRL/h** instead. This happens when the terminal program running on your desktop computer and the shell program on the UNIX computer are not in synch. You can usually fix these keys for the rest of your login session by typing one of the following lines at the prompt:

```
reset
stty sane
```

(That's short for "set teletype" – which gives you an idea of how long ago UNIX was created!). If these don't make it all good, try:

```
stty erase (backspace-key) [some keyboards call it backspace, some delete]
```

If the commands you type do not show on the screen at all, you may have accidentally frozen your screen by pressing **CTRL/s**. You can unfreeze it by pressing **CTRL/q**.

If you accidentally type something you didn't mean, you may find yourself in a program that you don't recognize. Don't panic! Here are some commands you might try entering, in recommended order (with return/enter after each):

q, Q, quit, exit, stop, logout

Here are some more (which do not require return/enter):

the Escape key,

CTRL/c, CTRL/d, CTRL/q, CTRL/x, CTRL/], CTRL/z

When you try to log out, you may get the message: `There are stopped jobs`. Simply type **logout** again, to kill those jobs and finish logging out.)

If all these commands fail, try closing the window; the UNIX operating system will then log you out automatically after a few minutes. If you're still stuck, try to get a consultant or your support provider to help you log out. If no consultant is available, as a last resort just reboot your workstation.

3.10 Printing files

If you run Windows or Mac OS on your workstation, the easiest approach is to download the file (as described in the next section), and then open the downloaded file with WordPad (in Windows) or TextEdit (in Mac OS), and print the file as usual from within that application.

It is quite common to end up with a messy file because the text is wider than the present margins in your document. One way to deal with this is to switch your printing to landscape mode, so that the text is printed lengthwise across the paper. To ensure that the document is really going to print out properly, it is useful to look at the document with the "Print Preview" feature.

3.11 Transferring files to or from research systems

The File Transfer Protocol lets you **put** the files that are on your computer on another computer, or **get** files from another computer. It is very simple to use. There are a few details that are discussed next. This section will focus on transferring files between computing systems; transfers to or from research data storage systems are covered in the section "Using research data storage systems" which follows later in this document.

3.11.1 From Windows

If your personal workstation has the Windows operating system, then use the SSH Secure File Transfer Client to transfer files. To transfer files in SSH Secure Shell for Windows, you'll first need to open a file transfer window. Double-click the SSH client icon on your computer, then, from the Window menu, select **New File Transfer**.

Once you have the file transfer window open, you'll need to connect to the computer you'd like transfer files to or from. Follow these steps:

1. From the **File** menu, select **Connect...**
2. In the **Host Name:** field, type the name of the host to which you are connecting (e.g., *quarry.uits.indiana.edu*). In the **User Name:** field, type your username on that host. Click

Connect. If this is the first time you have connected to this host, you will be asked if you want to save the new host key to the local database. Click **Yes**.

3. When you are prompted for a password, enter your password for that system.

When the software connects to your host, you should see your directories on that server on the left side, and the files in the active directory on the right.

To move a file from your computer to the server, follow these steps:

1. From the **Operation** menu, select **Upload...**
2. An **Upload - Select Files** window will open. Browse to the file you'd like to move, click on it to highlight (select) it.
3. Click the **Upload** button.

To move a file from the server to your computer, follow these directions:

1. Find the file or folder on the server you'd like to download, and click it to highlight it.
2. From the **Operation** menu, select **Download...**
3. A **Download - Select Folder** window will open. Browse to the folder where you'd like to place a copy of this file.
4. Click the **Download** button.

UNIX and Windows use different conventions for ending a line and beginning another, in text files. If you transfer a file to a UNIX box and the lines end with ^M, this is the problem.

Fortunately, file transfer programs usually recognize text files and automatically convert them. There is a utility named dos2unix which mends this problem: enter

```
dos2unix myfile mynewfile
```

3.11.2 From Mac OS

If your personal workstation is a Macintosh, we recommend that you use MacSFTP (available from IUware) to transfer files. Follow these steps:

1. Double-click the MacSFTP icon. The MacSFTP Login window will display fields for connecting to a remote computer.
2. In the **Host Name:** field, type the address of the remote host to which you wish to connect (e.g., *quarry.uits.indiana.edu*).
3. In the **Login:** and **Password:** fields, type your username and password for the remote host.
4. If you want to log into a directory other than your home directory, type it in the **Path:** field (e.g., *www*).
5. Click the **Connect** button to start the SFTP connection.

Note: The first time you connect to a host, SFTP may warn you that it can't determine the host's authenticity. Click **Yes** to accept the host's keys and continue connecting.

6. A window will open displaying the list of files on the remote host. To upload files or folders, drag them from a Finder window into the MacSFTP window. To download files or folders, drag them from MacSFTP into the Finder.

3.11.3 From UNIX

If your personal workstation has a UNIX operating system (including Mac OS X), you can use secure copy (**scp**) to transfer files. To copy *mydata* to your home directory on Quarry from the current directory of your workstation enter at the prompt:

```
scp mydata username@quarry.uits.indiana.edu:
```

You will be prompted for your password. Then you will see an indication that the transfer was successfully completed.

To copy a file named *file.dat* from your home directory on Quarry to the current directory of your UNIX workstation, enter at the prompt:

```
scp username@quarry.uits.indiana.edu:mydata .
```

The dot at the end is important (it means put it in the current directory, with the same name). Like the move command (**mv**), the copy (**cp**) and secure copy (**scp**) commands require both source and target parameters. You can change the name:

```
scp username@quarry.uits.indiana.edu:filename ./newfilename
```

Files may be transferred between supercomputers using scp or sftp just as they are from personal UNIX workstations, as described above.

3.12 Where should you keep your data?

Archival data storage is covered in the section "Using research data storage systems." You also need a place to store data while computing with it. Datasets up to one gigabyte may be kept in your home directory space, but should not be read or written repeatedly – that filesystem is not designed to take a heavy load. Such work should be done using shared scratch disk space. Incidentally, scratch space does not count against your storage quota (the default at IU is 10GB).

Global scratch space, */N/gpfs/*, is intended to be used as working space for running programs and temporary storage for program output. It's a high performance file system, mounted on all nodes. Local scratch disk space, */scratch/*, is available only to the particular node that it's associated with (a distinction pertinent to multinode jobs, also called "parallel programs").

To use scratch space, create a directory: enter

```
mkdir /N/gpfs/username
```

You can upload the data directly to that directory, either by browsing to it with a graphical-based version of scp or sftp, or (on a UNIX system) at the command line enter:

```
scp ./filename username@quarry.uits.indiana.edu:/N/gpfs/username/
```

Files are purged automatically from scratch space, to free up disk space as needed, so you need to maintain a copy in some other location (see the "Using research data storage systems" section of this document).

4 Running programs

To run a program, you just “say its name” to the computer:

```
$ date
Thu Aug 28 12:05:04 EDT 2006    [This is the computer's response]
```

4.1 Compiling FORTRAN, C, C++ programs on Quarry

You may only wish to use commercial programs such as SAS or Matlab. In that case, you can skip this section, but you still need to read the next one (“Batch processing”). In order to run a FORTRAN, C, or C++ program on any system, the source code must be compiled using that system's compilers, and *linked to that system's libraries*. You cannot normally copy a file from one computer to another, and expect it to run.

We recommend the Intel compilers on Quarry, named **ifort** and **icc**. You can copy a directory of example programs, and compile one with the Intel compiler, by entering the following commands:

```
cp -r ~hpc/simple_quarry_jobs .
cd simple_quarry_jobs
icc sine.c -o sine_c
```

To run this program, enter its name (the ./ means “in the current directory”):

```
./sine_c
```

The output will appear on your terminal.

4.2 Batch processing

Quarry uses Torque, a version of the “Portable Batch System” (PBS) to manage jobs (Big Red uses a similar system called LoadLeveler). If you have a program that runs more than 20 minutes, you must use the job management system to submit it for “batch processing” so that the system is used efficiently. There are two ways to do this: with “serialjob” (developed by Research Technologies staff), or with your own PBS submission script.

4.2.1 Serialjob

It is possible to run most single-processor commands as batch jobs by using serialjob – serial meaning single-processor – which writes your PBS script for you and submits your job. If the command that you wish to run is in your current directory, simply insert “serialjob” before the command. For example, to run the program named **sine_c**, then instead of

```
./sine_c
```

you enter the command

```
serialjob ./sine_c
```

If the program that you wish to run is not in the current directory, supply the full path to the command. Also, if it needs additional parameters, just include them. For example, to run a program named TheAnswer with data file data.dat, you could type:

```
serialjob /N/u/somedirectory/TheAnswer -d data.dat
```

Your program will be allowed to run for the default time allotted to jobs on the system. Serialjob is available on both Big Red and Quarry. For more information, enter **man serialjob**.

4.2.2 Submission scripts for the Portable Batch System

If you don't use serialjob to generate your "submission script," a file containing PBS directives, you must write your own.

You can use the "cat" command to examine a sample submission script for the file we just described compiling, which is included in the same directory of examples:

```
cat submit_sine_c.sh
```

which returns on screen:

```
#PBS -l nodes=1:ppn=1,walltime=5:00
#PBS -m ae
#PBS -q debug
${HOME}/simple_quarry_jobs/sine_c
```

This script consists of three PBS directives, which begin with #PBS, followed by the "execution line." The first line asks PBS for 1 node, and 1 processor per node, and 5 minutes 0 seconds of elapsed (wallclock) time to run it, once it starts. The second line tells PBS to email notification when the job aborts or ends. The third line sends this job to the "debug" queue, which is intended for short jobs only. Without it, the job would go to the "normal" queue. All PBS directives precede any other commands – in this case the fourth line, which is the executable command, entered just as it would be at the command line (that \$ isn't the prompt, it's a reference to the environment variable HOME, your home directory).

To submit a job to the PBS queue, use the **qsub** command:

```
qsub submit_sine_c.sh
```

The command returns the job's identification, or jobid:

```
480005.qm2
```

To check the status of your submission, at the prompt type:

```
$ qstat -u username
```

where **username** is your username. A table will appear that looks like this:

Job ID	Username	Queue	Jobname	SessID	NDS	TSK	Memory	Req'd Time	Req'd S	Elap Time
-----	-----	-----	-----	-----	-----	---	-----	-----	-	-----
480005.qm2	username	normal	--	--	1	1	--	0:05	Q	0:00

The status (S) shows Q (queued, waiting to run) or R (Running) or C (Completed). When your job has been completed, enter **ls** and you will see your program's output file, **job_sine_c.o480005**.

For more information, see the man pages or search the Knowledge Base for "PBS" or "Torque."

5 Big Red

To obtain an account on Big Red, IU researchers should visit the Account Management Service at:

<https://itaccounts.iu.edu/>

The information in earlier sections about using Quarry is almost entirely transferable to Big Red, so it will not be repeated here. In particular, the section “Where should you keep your data?” applies without change, since storage is distinct from computation.

This section focuses on the noticeable differences from Quarry. These mostly have to do with Big Red’s job submission system (LoadLeveler), and compilers (xlf, xlc, and xLC).

5.1 First time log in

To log in to Big Red, follow the instructions given above for Quarry, according to your desktop operating system. The hostname is **bigred.teragrid.iu.edu**. The different name, with "teragrid" instead of "uits," reflects the different purpose of this system. Big Red is an important national resource, which is dedicated primarily to large-scale analyses using parallel programs. This document makes no attempt to introduce parallel programming – various academic departments have courses for that. If you want your Big Red usage to be based upon a TeraGrid account, you should log in to **login.bigred.iu.teragrid.org**.

Your first login on Big Red brings up the firstshell script, just like on Quarry. You will be presented with the following greeting:

```
Welcome to Big Red!
```

```
This program is run the very first time you log in  
to Big Red to allow you to select your login shell.  
If you are uncertain which shell to select, choose  
ksh (Korn shell).
```

- 1) ksh
- 2) csh
- 3) bsh
- 4) bash
- 5) tcsh
- 6) quit

```
Select 1-6:
```

Unless you are an experienced UNIX user and have a specific reason to do otherwise, we strongly recommend that you select bash. Type **4** and press enter. Your change will be confirmed:

```
Your login shell was changed to the bash shell  
Select 1-6:
```

Type **6** and press enter to exit the system. You will receive the message:

```
Connection to is bigred.teragrid.iu.edu closed.
```

Once you select your preferred login shell it will be automatically set on the node you're currently logged in to. It might take up to 60 minutes to be propagated to all nodes.

Each time you login thereafter, you will see the login banner, or Message of the Day, which displays current announcements. It contains information regarding the status of the cluster.

5.2 Compiling

On Big Red the Fortran, C, and C++ compilers are named `xlf`, `xlc`, and `xlc++`, respectively.

Except for *very* large programs, compiling is easily done interactively. You can find some elementary examples in hpc's directory "examples." To view the contents of this directory, at the prompt type:

```
$ ls ~hpc/examples
helloWorld.C  helloWorld.f  llScript_C++
helloWorld.c  llScript_C    llScript_F77
```

The Fortran 77 example program is *helloWorld.f*, the C example is *helloWorld.c*, and the C++ example is *helloWorld.C*. To view the Fortran code, at the prompt, enter:

```
more ~hpc/examples/helloWorld.f
```

To experiment with these programs, copy the directory to your own home directory, *i.e.*, enter:

```
cp -r ~hpc/examples ~/
```

You may now compile your own executables. For example, at the prompt enter:

```
cd ~/examples
xlf helloWorld.f -o helloWorld_F77
```

This means "Run the IBM Fortran compiler, `xlf`, with input file *helloWorld.f*, and name the output file *helloWorld_F77*" (if the `-o filename` is not provided, `a.out` is always used). When it's done, you can run the output file:

```
./helloWorld_F77
```

5.3 Loadleveler

Any job which uses more than 20 CPU-minutes *must* be submitted to LoadLeveler. The script "serialjob" will write your Loadleveler submission script, as described in the chapter Running programs, above.

Sample LoadLeveler scripts are available in the `~hpc/examples` directory, named `llScript_F77`, `llScript_C`, and `llScript_C++`. To copy that directory into your own home directory, enter:

```
cp -r ~hpc/examples ~/
```

(Don't forget the `~` before `hpc`, to refer to hpc's home directory).

You can then use **cat** to view a script:

```
$ cd ~/examples
$ cat llScript_F77
#@ class = serial
#@ initialdir = ~/examples
#@ executable = helloWorld_F77
#@ output = helloWorld_F77.out
#@ error = helloWorld_F77.error
#@ queue
```

Note that the last line is “queue” – this is required. It is common practice to make a newly named copy of such a file, and edit it, when you have a new executable to run. For example,

```
$ cp llScript_F77 llScript_hostname
$ nano llScript_hostname
```

You can use an editor like Nano to edit `llScript_hostname`, to change the executable and output file names, as follows:

```
#@ executable = /bin/hostname
#@ output = hostname.out
#@ error = hostname.error
```

If you are using a TeraGrid account, you must include it:

```
#@ account_no = TG-STA123456N
```

To discover your TeraGrid account number, use the command

```
tgusage -u TeraGridUserName
```

If more than one account number appears, use trial and error to discover which one works.

To submit `llScript_hostname` to LoadLeveler, at the prompt type:

```
$ llsubmit llScript_hostname
llsubmit: The job "s10c2b5.dim.823708" has been submitted.
```

To check the status of your submission, enter

```
$ llq -u username
```

where *username* is your username. A table will appear that looks like this:

<u>Id</u>	<u>Owner</u>	<u>Submitted</u>	<u>ST</u>	<u>PRI</u>	<u>Class</u>	<u>Running On</u>
s10c2b5.823708.0	username	12/1 12:30	I	50	LONG	s15c1b12

The status (ST) shows an I (idle, waiting to run). Reenter "`llq -u username`" (perhaps several times) and you will soon see the status column showing an R. Your job is then running.

Eventually, reentry of "`llq -u username`" will show a C in the ST column, indicating your job has been completed. Now enter **ls** and you will see your program's output file, `hostname.out`.

```
$ cat hostname.out
s15c1b9
```

This indicates that the job, submitted from the node named `s10c2b5`, ran on node `s15c1b9`.

Some useful LoadLeveler commands are:

llclass	Describes the defined batch classes (queues)
llq	Queries the status of running and queued jobs
llstatus	Displays the status of LoadLeveler machines
llsubmit	Submits your script to LoadLeveler
llcancel	Cancels a running or queued job
showq	Shows information about job scheduling
showres	Displays job reservations

For more information about LoadLeveler and Maui commands, and several example scripts, search the Knowledge Base for "loadleveler" – the online documentation is at:

<http://kb.iu.edu/data/avim.html>

If your source code consists of more than one file, requires compiler options to improve run time, or if your research might benefit from improving the performance of your program, for example by parallelizing your source code, then please contact the High Performance Applications team.

5.4 Softenv

The Softenv environment management system is used on both Big Red and Quarry, to simplify environment configuration. When you login the first time, a file named `.soft` is created, setting up the default environment, such as compilers.

Additional packages may be added, if you wish. To temporarily add a keyword to your environment, enter:

```
soft add +keyword
```

To get a list of the possibilities, enter

```
softenv
```

To restore your environment to the default settings in the `~/.soft` file, enter:

```
resoft
```

To permanently change your environment, edit your `~/.soft` file. This can be done using Nano. For example, if you wish to run SAS and Matlab jobs, add keywords as follows:

```
# This is your SoftEnv configuration run control file.
# It is used to tell SoftEnv how to customize your environment by
# setting up variables such as PATH and MANPATH. To learn more
# about this file, do a "man softenv".
#
@quarry
+Matlab
+sas
```

For the changes to take effect, enter `resoft`, or log out and then log back in.

6 Commercial programs

The popular commercial programs SAS and Matlab are available on Quarry. As noted in the previous section, Quarry uses Softenv to manage the software environment. You must add the `sas` or `Matlab` keyword to your `.soft` file, to run SAS or Matlab, respectively.

Short jobs can be ran interactively, but for large jobs – those requiring more than 20 minutes of CPU time – you are **required** to use a job management system, as introduced above in the section titled "Running programs."

6.1 SAS

Sample SAS program and data files are available on Quarry. Once you are logged on, type:

```
cp ~statmath/scripts/sample.sas ~/scripts/  
cp ~statmath/scripts/sample.dat ~/scripts/
```

A SAS program normally contains a data file and a program file. For example, the code from the program file below reads a data file, such as `clas.dat`, with six variables: `id` (identification number of subject), `gender`, `wt_grp` (weight classification, 1=underweight, 2=normal weight, 3=overweight), `glucose` (glucose level), `bp` (blood pressure classification, 1=normal, 2=high), and `reactime` (reaction time for visual stimulus), from the root directory.

```
DATA sample;  
INFILE '~/scripts/sample.dat';  
INPUT id 1 gender $ 3 wt_grp 5 glucose 7-9 bp 11  
      reactime 13-15;  
PROC PRINT;  
RUN;  
PROC ANOVA data=sample;  
  CLASS gender;  
  MODEL reactime=gender;  
RUN;  
PROC GLM data=sample;  
  CLASS gender;  
  MODEL glucose=gender;  
RUN;  
PROC REG data=sample;  
  MODEL glucose=reactime;  
RUN;  
ENDSAS;
```

In the above example, the data file, `sample.dat`, is stored in the root directory of the user. If the data file is stored in another directory, then specify the full pathname, on the `INFILE` statement in the program file. For example,

```
INFILE '~/ingen/sample.dat';
```

tells SAS to find the data file and `sample.dat`, in the `ingen` subdirectory. The data file is stored in the directory in the following format:

```
1 m 1 99 1 210
2 f 2 320 2 420
3 f 2 195 2 350
4 m 1 110 1 215
5 m 2 218 2 364
6 f 3 120 1 355
7 m 3 125 1 335
```

6.1.1 Running SAS Jobs

Once you have both the program file and data file, to execute the job, you may choose one of the following three methods: interactive, background, or batch.

6.1.2 Interactive and background jobs

To run the program described above interactively, at the system prompt, enter:

```
sas sample.sas
```

Once the job has executed without errors, two additional files will be written to your directory, *sample.log*, and *sample.lst*. The *.log* file contains log information regarding your job, and the *.lst* file will contain the output listing. If the *.lst* file is not created, examine the *.log* file for any error messages. If there are error messages, edit the program file with corrections, and rerun the job.

However, if you run a job interactively, you may not be allowed to continue until the job is completed. The method of execution is only permitted only if your job requires less than 20 minutes to complete.

If you want to continue with other computing activities, while the job is running, then you can execute the job as a “background process” while the shell returns the system prompt. To execute the job as a background process, type a trailing ampersand:

```
sas sample.sas &
```

This is subject to the same short-execution-time constraint that purely interactive jobs are. The command “**jobs**” will list your jobs running in the background. Once the job is completed successfully, the *.log*, and *.lst* file will be written to the directory.

6.1.3 Batch jobs

If your SAS program will run for more than 20 minutes of processor time, or if you need large amounts of memory, you *must* use the batch system to run your program. The easiest way to do this is by using the command **serialjob**.

For example, if you normally run **sas sample.sas**, run **serialjob sas sample.sas**. The program **serialjob** will create an appropriate PBS or LoadLeveler script for you and submit it. Run **serialjob** from the directory that you want to be the current directory when the job actually runs; make sure that your SAS code and data are also there.

A second approach is to write your own PBS script, on Quarry, or Loadleveler script on Big Red. These scripts have been described above.

6.2 Matlab jobs

Again, there are three ways to run a Matlab program on Quarry: interactively, as a background process, or by submitting the program to PBS as a batch process. Any job, which requires more than 20 CPU-minutes, must be ran as a batch job.

6.2.1 Interactive and background processes

This is just typing the commands one-by-one at the Matlab command line. At the UNIX prompt you can start Matlab with the command:

```
matlab
```

Your terminal will display the Matlab "splash screen" and prompt:

```
      < M A T L A B >
      Copyright 1984-2003 The MathWorks, Inc.
      Version 6.5.0.196271 Release 13.0.1
      Mar 14 2003
```

```
>>
```

You can then enter Matlab commands (>> is the program's prompt). For example, to produce a four-by-four Hilbert matrix you can enter at the Matlab prompt:

```
hilb(4)
```

The program will respond with the following output:

```
ans =
    1.0000    0.5000    0.3333    0.2500
    0.5000    0.3333    0.2500    0.2000
    0.3333    0.2500    0.2000    0.1667
    0.2500    0.2000    0.1667    0.1429
```

To end Matlab and get back to the system prompt, enter the Matlab command **quit**.

For a brief introduction to the syntax and capabilities of Matlab, see the Stat/Math Center's introductory guide at:

<http://www.indiana.edu/~statmath/math/matlab/gettingstarted/>

You can also run Matlab in the background. To do this you first need to make a text file that has the commands you want Matlab to run, using a text editor such as Nano.

The following command will feed these commands to Matlab, from a file named *matlabinput*, and display Matlab's output on the terminal screen:

```
matlab < matlabinput
```

However, it is more common to have the output written to some other file. The following command will write Matlab's output to a file called *matlaboutput* and write any operation system errors to a file called *matlaberror*.

```
matlab < matlabinput > matlaboutput 2> matlaberror &
```

The ampersand at the end of the line tells the computer to run matlab in the background. After the program finishes running, you can look for files called *matlaboutput*, and *matlaberror*. The

file *matlaberror* file should be empty. There will also be a file called *results.mat* that records the results in a format Matlab can load later.

6.2.2 Batch processes

Any program that takes more than 20 minutes must be submitted to the job management system. To submit a job use the command **matlabjob**, which will create an appropriate PBS script (on Quarry), and submit it to the queue. You will need to specify the name of the input file using the command line option **-input**. For example, use:

```
matlabjob -input matlabinput
```

The output and error will be stored in files named from the input file containing suffices *.out* and *.err*, respectively. For, example if your input file is *matlabinput*, output will be in *matlabinput.out* and any errors will be in *matlabinput.err*. Optionally, you can specify names for the output and error files with the **-output** and **-error** options, respectively. For example:

```
matlabjob -input matlabinput -output matlaboutput -error matlaberror
```

A second approach is to write your own PBS script as described above.

For more information on using SAS, Matlab or any other statistical and mathematical computing software, contact the Stat/Math Center at 812-855-4724 or 317-278-4740, statmath@indiana.edu, or visit:

<http://www.indiana.edu/~statmath/>

7 Archival data storage

Each supercomputer is provided with multiple filesystems. Home directories (where you keep source code and small files) and scratch space (where you store data temporarily) were discussed above, in the section titled “Where should you keep your data?”. This section is devoted to long-term storage of “large” datasets.

The Research Storage group in the UITS Research Technologies division manages two systems, the Research File System and the Massive Data Storage System. You may use both, for different purposes. Which you should use is determined by your data's size and how often it is updated:

- The Research File System (RFS) is the right service if you need to store modest amounts of data, or data that must be accessed frequently. The initial quota is ten gigabytes.
- The Massive Data Storage System (MDSS) is designed for data archival purposes. MDSS is the right service if you require storage for tens or hundreds of gigabytes, or more, arranged in large files (at least 50 megabytes), that you access infrequently. The initial quota is one terabyte.

Files stored on the Research File System are stored on disk until deleted, so they are accessible quickly over the network. If you want to store your daily work files, or other frequently accessed data, on a system which is backed up regularly and which allows easy access, then the Research File System is for you. The Research File System is a scalable and robust storage system, accessible from a variety of environments at IU – for example from the central research systems (Quarry and Big Red), on-campus UNIX labs at IUB, or from your personal workstation (running Windows, Mac OS, or UNIX), as described later in this section.

The Massive Data Storage System (MDSS) is intended for users with projects that need large-scale archival storage. MDSS works best with large files, written once and seldom updated. Some examples include:

- Large write-once/read-many applications, like Geographic Information System files.
- Storage for experiments which collect large amounts of data, in a few large files;
- Simulations generating very large restart files, which need to be archived temporarily;
- Experimental metadata, stored for instrument calibration and data analysis.

In general, anything that's too large to fit onto available on-line storage, plus anything that is no longer needed on-line, but should be archived for possible future review, is a candidate for the Massive Data Storage System.

MDSS uses a hierarchy of storage devices, ranging from fast and expensive disk to slower but cheaper tape, to give the appearance of a single massive storage facility. When you send data to MDSS, it is written to both a tape and a "disk cache". Disk space is cleared as needed, with the most recently-accessed files kept on disk longest. If you later want to retrieve data is on tape only, a nominal delay is incurred while the tape is located and automatically mounted.

This is called *nearline storage* to distinguish it from *online storage*, which allows instant access to data stored in a computer's random access memory chips or on spinning disks. It takes 30-90 seconds, on average, to retrieve a data tape and load it into the robot's tape reader. Data is then read from the tape, written back to the disk cache, and delivered over the local network.

IU's data storage system was implemented with very long term goals. The hardware used by the Massive Data Storage System includes two large tape robots, one located in Indianapolis and the other in Bloomington. This geographically-distributed storage system provides tremendous data integrity, as it is possible to keep one copy of a data set in each location.

7.1 Requesting accounts

Once you have determined the storage service that best fits your needs, you may request a Research File System and/or Massive Data Storage System account. To request either MDSS or RFS accounts, go to

<https://itaccounts.iu.edu>.

To request an increased quota, send email to store-admin@iu.edu.

In order to establish a group, lab, or departmental account, you will need to request an appropriate network ID – also done at the IT Accounts web page. Once you have the group network ID and its password in hand, use it to log into the IT Accounts web page and request a storage account.

Undergraduates wishing accounts on the Research File System, the Massive Data Storage System, or Slashtmp must find a faculty sponsor for a specific research project and have the sponsor e-mail valid@iu.edu with reasons for the account request.

7.2 Accessing research data storage systems

7.2.1 Accessing the Massive Data Storage System with Secure FTP

If you are an IU student, faculty member, or staff member, you can download a secure FTP client (such as the SSH Secure File Transfer client for Windows, or MacSFTP for Mac OS) from IUware Online at:

<http://iuware.iu.edu/>

7.2.1.1 Windows

1. Click Start and select Programs, then SSH Secure Shell, and finally Secure File Transfer Client. If you are using a different SFTP client or if you moved your SSH Secure Shell folder after installation, then launch your client appropriately.
2. In the SFTP client window, click the Connect icon.
3. In the "Host Name:" field, type **sftp.mdss.iu.edu**. In the "User Name:" field, type your username. Click Connect, and supply your password in the dialog box that appears next.
4. When the software connects to your host, you should see a window with your directories on your local computer on the left side, and your directories on the remote host on the right.

To move a file from your computer to the Research File System server, follow these steps:

1. From the Operation menu, select Upload ...
2. An Upload - Select Files window will open. Browse to the file you'd like to move, and select it.

3. Click the Upload button.

To move a file from the server to your computer, follow these directions:

1. Find the file or folder on the server you'd like to download, and click it to highlight it.
2. From the Operation menu, select Download.
3. A Download - Select Folder window will open. Browse to the folder where you'd like to place a copy of this file.
4. Click the Download button.

7.2.1.2 Mac OS X

1. Double-click the MacSFTP icon. In the "Host Name:" field, type the address of the remote host to which you wish to connect (e.g., **rfs.iu.edu** or **sftp.mdss.iu.edu**).
2. In the "Login:" and "Password:" fields, type your username and password for the remote host. You have the option of saving your password to the Keychain. If you want to log into a directory other than your home directory, type it in the "Path:" field.
3. Click the Connect button to start the SFTP connection. (Note: The first time you connect to a host, MacSFTP may warn you that it can't determine the host's authenticity. Click Yes to accept the host's security keys, and continue connecting.)
4. A window will open displaying the list of files on the remote host. To upload files or folders, drag them from Finder windows into the MacSFTP window. To download files or folders, drag them from MacSFTP into the Finder.

7.2.1.3 UNIX/Linux systems (for example, Big Red)

You can use **sftp** to copy files from the Massive Data Storage System (it's already installed). You will probably want to create a directory in the scratch directory, to get better performance and avoid exceeding the quota in your home directory. The commands to use are:

```
mkdir /N/gpfs/username  
cd /N/gpfs/username
```

Once in the correct directory, start the sftp program and connect:

```
sftp sftp.mdss.iu.edu
```

You will then be asked for your password, and presented with the sftp program's prompt, at which you may enter commands, for example:

```
sftp $ help
```

Within the sftp program, you can list the files in the remote directory (with the **ls** command), change directories on the remote system (with **cd**) or on the system where you started sftp (with **lcd**, "local change directory").

To quit sftp, enter **quit**.

7.2.2 Mounting the Research File System on your desktop system

Once your account is created, you can access your Research File System files and folders as if they resided on a drive on your workstation.

In **Windows**, to access your RFS folder directly:

1. Right-click **My Computer** and select **Map Network Drive...**
2. In the "Folder:" field, type the path for your RFS account (replace *username* with your username):
`\\rfs.iu.edu\username`
3. Click **Finish**.

You will be prompted to enter a username and password. Your Research File System folder will then be mapped to a drive on your workstation.

In **Mac OS X** you can mount your RFS space as follows:

1. In the Finder, from the **Go** menu, select **Connect to Server...**
2. In the Connect to Server window, in the "Address:" field, type:
`smb://rfs.iu.edu/username`
(replace *username* with your username). Click **Connect**.
3. You will be asked to authenticate. In the Workgroup/Domain box, type **ADS**. Type your username and password in the appropriate boxes. Press Return.
4. Click **OK**. Your Research File System folder will be mounted on the desktop shortly.

In **Linux**, the system administrator can establish file system access to the Research File System with the following commands:

```
mkdir /rfs  
mount -t smbfs //rfs.iu.edu/username /rfs
```

7.2.3 Hierarchical Storage Interface

Hierarchical Storage Interface (HSI) and its companion program, HTAR, can simplify aggregation of many files into one large file, which is the preferred method of storage in HPSS. HSI commands will seem familiar to Unix and FTP users.

On the Research Database Complex, HSI is installed in `/usr/local/bin` (so it's in the default path). On Big Red and Quarry, you may need to add `+hpss` to the end of your `.soft` file, and give the `resoft` command to make it available (you need do this only once). To use HSI on your personal workstation, download a copy from the MDSS home page.

A session might look like the following (% is the Unix shell prompt, and ? is the HSI prompt):

```
% $ hsi  
Principal: jdoe  
[jdoe]Password:  
Username: jdoe UID: 11021 CC: 11021 Copies: 1 [hsi.3.3.3 Fri  
Jan 12 13:36:06 EST 2007]  
  
% ? ls  
/hpss/j/d/jdoe/:  
NPB-ppcc.tar foobar/ movies/
```

```

? du -k
861309  /hpss/j/d/jdoe/
-----
861309  total 1024-byte blocks, 6 Files (881,979,719 bytes)

? put myfile1.dat
put myfile1.dat : /hpss/j/d/jdoe/myfile1.dat ( 10485760 bytes,
12283.4 KBS (cos=3))

? cd movies
? get myfile2.mov
Scheduler: retrieving file(s)
get myfile2.mov : /hpss/j/d/jdoe/movies/myfile2.dat
(2005/09/29 08:49:03 10485760 bytes, 16842.8 KBS )

? exit

```

You can find more information at

http://storage.iu.edu/hsi_docs/

8 High speed temporary storage

We have mentioned several types of storage: home directories, local and global scratch space, and archival storage. These exist at most computing centers. Created in part to meet the demands of modern electronic sensors, the Data Capacitor is a high-speed file system, which has achieved world-record simultaneous read/write performance (approaching theoretical limits).

The Data Capacitor is not intended for permanent storage of data, and is not backed up. You can archive data stored on the Data Capacitor on IU's HPSS using, for example, HSI or any of the other methods used to access HPSS. It is your responsibility to arrange for long-term permanent storage of any data on the system as needed.

The Data Capacitor is mounted on Big Red and Quarry at the mount points `/N/dc/` and `/N/dcwan/`, where it behaves like any other disk device. All Big Red, Quarry, and TeraGrid users have a scratch space directory, available at `/N/dc/scratch/username`. Space is not allocated, so the amount of storage varies, but terabytes are available. You can access your scratch space at `/N/dc/scratch/username`.

Users can also request long-term project space. The default allocation is 10TB. Projects may be awarded more than 10TB if justified. Access to allocated project space is available at `/N/dc/projects/projectname`.

Users at other institutions (including IU researchers with remote accounts) can request Data Capacitor storage space, which can be mounted at the remote institution, as well as on Big Red and Quarry. Access to the WAN space is available at `/N/dcwan/`

9 Using IU's visualization facilities

In addition to understanding your analysis and how to best represent your data visually, it is important to consider how visualization tasks factor into your scientific workflow. In the most common scenario, you simply use a workstation on your desktop or in your lab to run visualization software, be it commercial, open source, or custom-coded. Your data may reside on the local file system or a mounted file server, but all computation and visual display occurs on the local system.

In some instances, your visualization and analysis work may require the use of special resources available via the network, such as retrieval of large amounts of data from the mass store system, parallel computation from a supercomputer, or access to remote hardware or instruments. Working in conjunction with other parts of UITS, Research Technologies visualization staff can help design a system that maximizes your efficiency.

In other instances, your visualization task may benefit from special purpose hardware, or computing systems with more memory or processing power than your local system. In such cases, you are invited to use the resources of the Advanced Visualization Laboratory (AVL), in the ICTC Building on the IUPUI campus and in Lindley Hall on the IUB campus.

For more information see:

<http://avl.iu.edu/>

9.1 Supported visualization platforms

Our primary supported visualization platforms are Windows and Linux; however, we can also provide limited support for Mac OS X and IRIX. Whenever possible, we attempt to identify and adopt tools that are platform independent; however, special-purpose displays, hardware cards, and software packages may impose restrictions.

9.2 Visualization services

Software consulting - The most effective way to get started with visualization (or to enhance your current work with visualization) may be to arrange a consultation with our staff (send email to vishelp@indiana.edu). The AVL staff may be able to help you derive more functionality out of the applications that you are currently using, or to help you identify better tools for your needs. They can also assist with data translational tasks, and application optimization through custom scripts, macros, or plug-ins.

Custom software development - There may be no satisfactory software tools for your visualization task. In such cases, the AVL may be able to partner with you to develop a custom application that precisely fits your needs. While such developments may take several months or more, the result can be extremely valuable. The AVL has developed custom applications for volume rendering, molecular, phylogenetic and pedigree tree, and information visualizations.

Hardware consulting - While central facilities are satisfactory for occasional use, one-time demonstrations, or technology "test drives", visualization technologies have the greatest impact

when they are conveniently and directly accessible in the user's lab or department. Our staff can help you to derive specifications for the system or systems that best fit your needs, including desktop workstations, specialized hardware, and displays. We also have good working relationships with several vendors should that be necessary.

9.3 Visualization software

Identifying, exploring, and providing access to the latest and most advanced visualization software is a key goal of the AVL. As such, we constantly conduct extensive investigations into a wide range of frameworks, APIs, and software tools that may prove useful for the visualization needs of the University community. While we generally prefer open-source or free tools, there are situations that are more appropriately handled with proprietary tools. We host our own license server and are able to provide a limited number of licenses for key visualization software packages that include (but are not limited too) VirTools, CAVELib, Maya, 3D Studio Max, Rapidform, and ZEdit Pro. Open-source, free, and/or locally developed codes include (but are not limited too) VTK, ParaView, X3D, 3DIVE, PathView, XMView, Effect, and OpenGL. Our list of recommended software is vast and ever-changing. We encourage you to schedule a consultation with an AVL team member to ensure you are using the most appropriate software tool for your needs.

9.4 Visualization facilities

While many visualization tasks can be carried out on desktop workstations, some tasks may have highly specialized hardware or display requirements, or may have computation or memory requirements that exceed those of your desktop system. For these situations, the AVL maintains central facilities on both the Bloomington and Indianapolis campuses that provide a range of advanced technologies including (but not limited to) virtual reality and high resolution displays, advanced interface/input/output devices, the latest GPUs and rendering hardware, and a selection of teleconferencing and tele-collaboration systems.

For a specific listing and more detailed information see:

<http://avl.iu.edu?facilities>

10 Contact information

General purpose computing (Windows, MacOS X, Linux and Unix user support): contact the Support Center by phone at (317) 274-4357 [IUPUI], (812) 855-6789 [IUB], by email to ithelp@iu.edu or visit:

<http://kb.iu.edu/>

The UITs Research Technologies division is a part of the Pervasive Technology Institute:

<http://pti.iu.edu>

Biological computing:

Send email to bioinfor@iu.edu or visit:

<http://kb.iu.edu/data/alei.html>

Biomedical applications:

Send email to data@iu.edu or visit:

<http://biomedapp.iu.edu>

For MySQL and Oracle *research database* issues, send email to hps-admin@iu.edu.

Data storage:

Send email to Research Storage at store-admin@iu.edu or visit:

<http://storage.iu.edu/>

Digital library development:

Send email to the Digital Library Program at diglib@indiana.edu or visit:

<http://pti.iu.edu/dlp/>

Programming and parallel algorithm development:

Send email to High Performance Applications at hpa-admin@iu.edu or visit:

<http://pti.iu.edu/hpa/>

Statistical and mathematical computing:

Call the Stat/Math Center at 812-855-4724, send email to statmath@indiana.edu or visit:

<http://www.indiana.edu/~statmath/>

Systems administration:

Send email to the High Performance Systems group at hps-admin@indiana.edu or visit:

<http://pti.iu.edu/hps/>

Visualization:

Send email to the Advanced Visualization Lab at avlstaff-l@indiana.edu or visit:

<http://avl.iu.edu/>

A final note...

These notes become outdated and must be revised frequently. Your insights are invaluable, to help us improve the information that will be offered in the future. *Was anything missing? Was anything unclear? Was there more than anyone needs to know?*

We take your advice seriously. Please send your comments to researchtechnologies@iu.edu.

Acknowledgments

We gratefully acknowledge the support of the National Science Foundation, the Lilly Endowment, Inc., the National Institutes of Health, IBM, Inc., StorageTek, and Sun Microsystems; as well as support from the State of Indiana and Indiana University itself. Without the generosity of all of these organizations, we would not be able to offer the systems and services discussed in this document. The support of the Lilly Endowment, Inc. for the Pervasive Technology Labs, the Indiana Genomics Initiative, and METACyt have been particularly important in advancing IU's strategic goals in advanced information technology and life sciences. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of those organizations.

Contributing authors

Andrew Arenson
Michael Boyles
Robert Cruise
Arvind Gopu
David Hart
Peg Lindenlaub
Mary Papakhian
John Samuel
Kurt Seiffert
Anurag Shankar
Craig Stewart
Eric Wernert

Editorial assistance

Malinda Lingwall
Paul Messing
Sean Pendergast