Welcome to *Biostatistics!*

Introduction:

This series of worksheets, available online at: http://biotoolbox.binghamton.edu/ or as a booklet from the University Bookstore, comprises lecture notes and lab manual for Biology 483N/597 - an introduction to the field of *Biostatistics*. The primary objective of this course is practical application of statistical methods to solving biological (and other) problems. As a result, we will not be so much concerned about background theory or other matters that might mostly be of interest to mathematicians. There are courses in the Mathematics Department that handle that. Instead, the focus here is on doing useful work in a structured and disciplined way that promotes confidence in statistical results. The notes presented here are to a degree intended to stand alone. However, they are also meant to be used with the classic textbook: *Biostatistical Analysis 5th edition* by Jerrold H. Zar Prentice Hall (Pearson) 2010 and the above website.

By "classic" I mean that Zar's textbook as well as many others, such as excellent texts by Rosner and Sokal & Rohlf, present statistics as a narrative of "first principles". They typically feature mathematical arguments in the main part of the text accompanied by small "cooked" examples showing how calculations may be accomplished by hand. That's the way statistics has traditionally been taught and there's still much to recommend working examples in this time-honored fashion. However, no one in their right mind should consider working real-world problems this way - even with the help of ubiquitous and powerful MS Excel. In this day and age, there's definitely a better way!

Proprietary and largely stand alone statistical computing packages, such as Minitab, SPSS (formerly PASW, now IBM), STATA, StatPlus, SYSTAT, SAS, XLSTAT, and others remain very popular. Important statistical tests are typically well-implemented using intuitive Graphical User Interfaces (GUI), and most programs are capable of handling analysis as well as graphing of large datasets. However, this approach comes at a considerable cost, not only in terms of dollars (often thousands!), but also in terms of adaptability and portability. If you leave the university setting where these programs have been site licensed, you may be out of luck. Also, in real-world biological research, one may become involved in a collaboration with researchers at other institutions, perhaps even on a different continents. In such working groups, invariably some individuals will be using PC's and others Unix machines or Macs. Although portability of data between platforms and proprietary programs is usually not difficult, there often remains the very real problem of understanding what members of the research team may be doing. Using different proprietary programs, it is often not clear that results derived from one member is necessarily compatible with another. This is especially the case when, as is likely to occur, particular researchers may be uncertain about how to configure a large set of options that typically inhabit a specific GUI. In these circumstances, much more time may be wasted trying to figure out the oddities of the GUI than actually doing statistical analysis. Added to this is the fact one or more members of a research team may wish to try some newly published statistical technique. New methods originate all the time, but it may takes years and significant additional dollar cost, before they become commercially available.

In the past few years, global statistical computing has come of age! In particular, the R Project for Statistical Computing at: http://www.r-project.org/ is in the process of revolutionizing the way many biologists conduct their statistical research, and research teams interact with each other. Managed by an international steering committee, the R project is an open source statistics oriented programming language that is completely free of charge. Unlike most freeware, however, R is exceptionally powerful and easily extended by means of "packages" offered by high-end users. Thus, it often offers the newest and best methods available at any price. The only downside about R is that it is script based. That is, one has to write out a little program of instructions telling R what to do. Rudimentary GUIs, such R commander (see: www.jstatsoft.org/v14/i09/paper), are available for R as packages, and a reputedly spiffy commercial GUI is available for the closely related language S-plus at: http://spotfire.tibco.com if you really wish to pay for one. However, once you get used to it, writing scripts is easy. Moreover in collaborative work, R scripts can be sent along with data to colleagues who also have free access to R, so everyone can be certain that each is running an absolutely identical analysis. If an approach changes, all members of the group have to do is modify and resend the script!

Purpose of these Worksheets:

The purpose of these worksheets is to ease the transition between what Zar and others write in textbooks, and useful implementation of specific concepts or methods as a script in R. There are a couple important issues in doing this:

1) Developing a professional approach to the use of statistical software:

As one moves away from old-fashioned hand calculations to use of modern software, the issue of what to do for oneself versus acceptance of computer generated results becomes an important problem. Much too often, I observe both students and mature scientists preferring to implicitly believe output derived from unknown calculations somewhere in a software's "black box". Of course, for professional work, this is very dangerous. Also, when faced with different results from more than one "black box" even more silly, but unfortunately common, is a misguided form of consumerism:

> **"My software cost more, so it must have been better tested..." or "My software is what the professor or advanced graduate students use in the lab, so it must be better..." or "This is the software everyone has always used in the lab and I have a cheat sheet, so I dare not change it..."**

As in many areas of science, difficulties in use and understanding of statistical software can easily be resolved by adopting a thoughtful strategy and reasonably disciplined approach to the problem. To conquer the "black box", *I cannot recommend strongly enough* the importance of what I call the "prototype".

A prototype is a fully worked and completely annotated example.

Before attempting use of any software to answer a research problem, it is important to test the software first! **Testing usually means using the software to answer a statistical problem *for which one already has the answer*. By determining that an answer is correct, as verified by another source such as a worked textbook example, then one has at least some confidence about what the "black box" is doing. Moreover, if one calculates at least some aspects of the problem by hand and carefully labels interim results, then the prototype may be used to identify other important information each "black box" typically provides. In my opinion, *no one* should feel that they have *earned the right* to do useful work with any statistical software, no matter how well it comes recommended or cheatsheeted by others, *without constructing a personal prototype first.* Although I commonly encouter individuals in too big a hurry to bother with making prototypes, I argue that over the long run, prototypes actually end up saving a huge amount of time. Once made, they are useful practically forever. When returning to a test that you haven't run for some time, you don't have to start over relearning everything from the beginning.**

The following worksheets are *my* prototypes of many procedures commonly encountered in a first-level *Biostatistics* course. They are organized by a numerical code written in the header of each worksheet corresponding more-or-less to the way topics are presented in Zar. Each worksheet typically begins with a general statement of the problem along with reference to one or more examples analyzed in the prototype. The original data can be found online in the appropriate section of the Biotoolkit website, and may be downloaded for use in constructing your own prototype. Following this, the worksheet performs as many hand calculations as possible that may be directly compared with worked examples in Zar's text. Following this first working of the problem is a section labeled "Prototype in R". In this, you will find annotated portions of an R script that also works the problem successfully, along with annotated portions of R's output. The full R script is also available for use online in a text format that allows direct cutting and pasting into the R interpreter. For those who may use the booklet form of these notes as a stand alone resource, R scripts are also repeated in the final section.

These worksheets are offered for your benefit, but in themselves *do not satisfy work on your part!* As part of this *Biostatistics!* course you will be required to set up a portfolio of professional-level prototypes of your own *done your way* and in a format *you will remember!*

2) MathCad is mostly beautiful, but R script may look ugly:

By this, I mean that R scripts are written in a powerful and highly evolved computer language involving complex rules of syntax. There are typically several ways to program a particular procedure in R some more-or-less easy to read, others decidedly not. However, rarely does the syntax look like calculations shown in the text, nor is it often easy to see how the classical textbook equations actually relate to the R script. To ease the transition, each of my prototypes is implemented in an engineering calculation software program called MathCad. For details about MathCad, see: http://www.ptc.com/products/mathcad/. I make my prototype files available online in both a "static" form written as Adobe *.pdf files and in a MathCad native form *.mcd that will run all calculations "live" if you own MathCad. If interested in purchasing the program be sure to look for the subtantial student discount. Otherwise, the static view is typically more than sufficient.

MathCad is similar in function to perhaps more familiar MatLab, see: http://www.mathworks.com/, in providing much greater power and flexibility in calculation than a program like MS Excel. However, MathCad is also "beautiful' in that it allows me to use notation that looks very much like Zar's equations while at the same time providing accurate results of calculations based on example data. In general, there's little to worry about in reading my MathCad prototypes as long as one realizes that each formula in the live version actually performs a calculation, and that overall flow of the worksheet runs from left to right in each line and from top to bottom in the worksheet overall. Occasionally a section of a worksheet will include a set of MathCad instructions designed to display a graph or assemble intermediate calculations that you may not find very interesting. That's OK, you can skip over them. In addition, following standard practice in computer programming, there are two distinct meanings to the traditional concept of "equals".

For instance, I can easily set up a vector (i.e., a list) of numbers by using the *assignment* meaning of equals :=

$$V := \begin{pmatrix} 1 \\ 3 \\ 5 \\ 8 \end{pmatrix} \qquad \text{mean}(V) = 4.25$$

and then proceed to work on it, such as taking the average using MathCad's built in function mean(). The result is then output using the *valuation* meaning of equals =

Note the flow of this line in the worksheet, and that the notation := and = are different!

Matrix calculations are equally easy, such as:

$$M := \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

$$M \cdot M = \begin{pmatrix} 30 & 36 & 42 \\ 66 & 81 & 96 \\ 102 & 126 & 150 \end{pmatrix}$$

If you are unfamiliar with matrix algebra, don't worry about what this means. However, you can see how concise and much like a textbook presentation this calculation (involving sums of products of rows versus columns) appears to be!

In all worksheets, be sure to look for the distinctly different *assignment* versus *valuation* meanings for equals, and following the sheets should be straightforward. Also look at Zar's "hand" calculations, to compare results. The basic point of each prototype is to show how the mathematical equations in Zar are related to his "hand" calculations while at the same time reporting accurate numerical results from the example. From this, one should be able to identify important numerical values representing intermediate steps in the calculation that statistical software output may report in one way or another.

As intuitive and beautiful as MathCad may be as a bridge between textbook and computer-based programming, for useful statistical work involving complex calculations the R language nevertheless has it beat by a mile! No doubt, you will want to build your own portfolio of prototypes using properly annotated R scripts. Once constructed, these scripts will be little gems that you will want to keep! With them, combined with mastering basic skills in modifying scripts as different cases dictate, you can effortlessly run any procedure anytime with practically any kind of data, and with confidence that you do, in fact, understand the result. To get started, see worksheet Biostatistics 801 in the booklet, also "Getting Started with R" online.