A Template for Online Homework: Frankenstein's Monster or Robo TA?

Roger A. Dahlgran[1]

Abstract

This paper describes the programming procedures required to implement online homework and evaluates the application of these procedures based on use in the author's course. The description of the procedures utilizes a template showing two representative applications from the author's introductory econometrics course. In one, the students are to collect and record data and in the other, students are to perform econometric analysis on the data. The web address for the template is arec.arizona.edu/RoboTA. The use of online homework in the author's economics of futures market course revealed that the benefit-cost tradeoff is between the savings of instructional time spent grading homework and increased instructional time spent developing homework assignments. Online homework is favored by (1) large class sizes, (2) numerous, difficult to grade, numerical homework assignments, (3) continuity of these conditions, and (4) the availability of adaptable programming solutions. Online homework was not found to be more effective in helping students learn, though in some instances automation can lead to more assignments and these additional assignments can be beneficial. The implementation of online homework was associated perceptions of greater usefulness of the course web site and lectures.

Paper presented at the Western Agricultural Economics Association Annual Meetings, Long Beach, California, July 2002.

[1] Department of Agricultural and Resource Economics, University of Arizona, Tucson, AZ Phone 520-621-6254. FAX: 520-621-6250. E-mail: dahlgran@u.arizona.edu.

A Template for Online Homework: Frankenstein's Monster or Robo TA?

The Internet potentially affords sweeping changes in teaching methods for all college-level subjects (Ramstad). Some of these new teaching methods are applicable to agricultural economics instruction. In addition, the emergence of these new teaching methods is of interest to agricultural and resource economists because it presents the opportunity to study technological change and adoption at close range, in our own instructional programs. Like technical change in other sectors, technical change in instruction derives from new inputs that are either less costly or more productive than existing inputs. While most economics instruction is reported to utilize the large lecture format (Becker, 1997) beneficial Internet applications have been reported (Vachris, 1999, and Agarwal and Day 1998). Agricultural and resource economists also find the Internet to be a useful instructional tool as a recent survey found that 45 percent of the undergraduate agricultural and resource economics courses taught in the western U.S. had an Internet presence (Dahlgran, 2002). Half of these courses used the Internet to convey subject matter while web sites for the other half are limited to course syllabi and instructor contact information.

A natural progression following the establishment of a course web site is to process numerical homework assignments online. The stylized model of this application is that students access homework problems and submit responses over the Internet. A program on the server evaluates the responses and provides immediate feedback to the student. While such interactive applications are widely used on the Internet, they were rarely found on agricultural and resource economics course websites (Dahlgran, 2002).[2]

In addition to reducing the grading workload, one advantage of online homework is that it permits beneficial instructional practices that are too costly to implement under traditional methods. For example, online homework can be iterative if the student is given the opportunity to refine his/her thinking and continue working on the problem based on the feedback received. This method exercises students' active experimentation learning mode (Kolb, 1984). Second, the online approach can be used to present each student with a unique problem generated either randomly or by systematic rules that are not apparent to the student. This increases the likelihood that each student submits original work. Third, submittal deadlines, required to efficiently utilize the instructor's time in batch grading traditional assignments, are not necessary with online homework. As a result, each student can select the best pace for study.

The difficulty with implementing online homework is that the computer programming is complex requiring, like Frankenstein's monster, the stitching together of pieces from many different bodies of programming concepts. Concepts such as HTML, JavaScript, CGI, Perl, SQL, ODBC and OLE must all be used. Most instructors do not have knowledgeable computer programmers at their ready disposal and substantial time is required for the instructor to learn the programming concepts and write code. However, once the procedure is developed, its implementation resembles Robo TA, willing to endlessly grade homework and provide feedback, costing nothing to maintain, patiently accommodating the trying and testing learning style for

[2] Other applications of the Internet for communications surrounding homework were found. These include using the Internet (1) to make assignments available to students, and (2) to collect responses via email. Our model is more comprehensive than these applications.

large numbers of students, and generating individualized problems under a wide variety of assumptions. Also, once developed, the procedure is readily adaptable to other interactive instructional applications such as grade reporting to students, simulations, and online testing.

The purpose of this paper is twofold, first to describe programming solutions required to implement online homework, and second to evaluate this approach based on its implementation in the author's courses. In describing programming solutions, alternative approaches will be generally described, a specific solution will be discussed in more detail, and a web address where working programs can be inspected and downloaded by the interested reader will be revealed. Online homework will be evaluated based on responses from a student survey regarding satisfaction with this approach and based on examination results.

Methods

Traditional homework assignments follow these steps: (1) the homework is distributed, (2) the student records answers, (3) the answers are collected, (4) the assignment is graded, (5) the grade is recorded, and (6) the homework with feedback is returned to the student. Automating this process requires a course web site from which the student can retrieve the homework assignment (1). This document, composed of Hyper Text Markup Language code (HTML), must contain a HTML form that contains input fields for text or numbers or other input mechanisms such as dropdown menus, radio buttons or check boxes. The student responds by recording answers in the input fields or indicating choices (2). When the homework is completed to the student's satisfaction, the student passes the form to the server by clicking the form's "submit" button (3). The server extracts the student's responses, processes them, and computes a score based on discrepancies between the student's responses and the correct answers (4). The student's score is recorded in a database on the server (5). Finally, the results are communicated back to the student (6).

While an understanding of HTML code is required to implement these procedures, HTML is not the focus of this paper. Instead we focus on programs that present assignments, collect and evaluate student input, record the results, and report back to the student. Accurately identifying the user during the interaction is a concern, especially for large classes, so programs for site-wide login and for account maintenance are also considered. The account maintenance programs provide for self-registration, password revisions, name and email address editing, and password recovery via email.

These programs are demonstrated and the code can be downloaded from http://arec.arizona.edu/RoboTA/HWMenu.htm (figure 1). This page serves as a template for online homework in any course website. While the source code would not be revealed to students on an actual course website, it is provided here so that other instructors can use or modify it to achieve other instructional objectives. Potential modifications include enforcement of submission deadlines, prevention of repeated submittal, and provision of session resumption. The first two sections of the template respectively demonstrate procedures for online homework and user login account maintenance.

Interaction with server-based databases is the common feature of the programs. This interaction permits the student to transparently add, edit, and delete records. Two popular programming methods for achieving user-specific server interaction are the more traditional common gateway interface (CGI) (Metzler, and Michalski, 2001; Guehlich, et. al., 2000), and the newer Microsoft active server page (ASP) technology (Mercer, 2001; Reselman, 2000; Buser, et.

al. 1999). The primary difference between these two methods is that with ASP program variables are embedded in the HTML document while with CGI the server runs a program that generates the HTML. These approaches use different programming languages. Perl is a public domain scripting language (available as a free download from www.activestate.com) widely used for (Wall, Christiansen, and Orwant, 2000; Schwartz and Phoenix, 2001), while ASP uses server-side VB Script, which is much like Microsoft's Visual Basic language. Another difference is that the CGI method works on both Microsoft NT and UNIX servers while the ASP method is limited to Microsoft NT servers. The CGI method is outlined here.

The hyperlinks in the homework menu template (figure 1) run various programs. We will briefly summarize the operation of each leaving the detailed study of individual programs to the critically interested reader. The first section of the template models a list of homework problems that might be part of a course. The examples used are from an introductory graduate-level econometrics course (http://arec.arizona.edu/econ418/). These problems are especially well suited to Internet submission and grading because numerical answers are required and correct answers are found only through mastery of the computational procedures taught. Online homework has also been used successfully in an economics of futures markets course with an average enrollment of over one hundred students. Homework problems in this course involve security pricing, options pricing, currency arbitrage, and cross-hedging (http://arec.arizona.edu/arec313/).

Assignment 1 is to collect and record data for use in regression analyses in subsequent homework problems. The assignment 1 hyperlink executes the Perl program hw1.pl, which outputs the HTML that is rendered on the student's monitor (figure 2). Normally the student enters the data then clicks the Save button. If the student has no data in the MS Access database, then the recorded data are added to the database.[3] If the student has previously saved data, then these data are overwritten with the data from the form. If the student clicks the load data button, the Access database is queried for records identified with the student's ID number and the input form is generated with the retrieved data in the respective input fields. If the student clicks the clear data button, all input fields are set to blank and the page displaying blank values is presented.

The MS Access database is accessed using the open data base connectivity (ODBC) standard and the Win32 OLE (object linking and embedding) Perl module (Roth, Dubois). To retrieve records from the Access database, an ODBC data set name (DSN) must be previouly defined. Perl connects to this ODBC DSN, queries the connection using the Access structured query language (SQL), and manipulates the records returned. New records are written with the AddNew command and existing records are updated with the UpDate command. Both of these commands are part of Perl's Win32 OLE module.

For assignment 2, the student uses the data collected in assignment 1 to compute the theoretical variance of the estimated regression intercept, slope, fitted values and predicted values for various x values (figure 3).[4] The HTML document for the assignment contains the student's data, instructions, and input fields for recording the computed variances. Clicking the "Check Your Answers" button computes the student's current score. The student can continue working

[3] These examples all use tables in a Microsoft Access. Any other program that shares supports the ODBC standard could be used instead. With additional programming, text files can serve as a database.

[4] The answers are couched in terms of σ^2 times c. As σ^2 is unknown, the student must compute c based on the values of x_i.

in an attempt to increase this score. When the student is satisfied with the work, clicking the "Submit Your Answers" button causes the score to be computed and recorded in the grade book (another Access database table). Either button causes the program to re-execute with the procedures selected based on the button clicked.

Assignment 2 illustrates exploitation of the server's computational capabilities. Each student's data are unique so each student's answers are unique and the computations required to derive the answers are intricate. Manually grading this assignment would be prohibitively time-consuming but once the computational procedure is coded, grading requires no time. The different samples vividly illustrate the notion of sampling distributions of regression statistics. This illustration is not available with traditional homework assignments where all students have the same data. In this case, online homework enriches the presentation of the concept.

Assignment 2 utilizes the login procedure. This procedure checks the student's login status by determining whether a valid cookie for the web site exists on the student's computer (figure 4). If the cookie exists, then the user is already logged in so the cookie's expiration time is extended and login procedure terminates being invisible, so far, to the user. If the cookie doesn't exist, then the student is prompted for a user ID and password. The password table is queried for a matching user ID and password. A match indicates a valid user so a cookie is placed on the student's computer. If a user ID and password match does not occur, the student is re-prompted for a valid user ID and password. The login program code can be called either as a subroutine from within a homework exercise or as an independent program from a hyperlink as in the second section of the menu.

The second section of figure 1 provides account maintenance (logon, logoff, create new account, delete existing account, edit account data, and retrieve forgotten password). The code that performs these tasks demonstrates more generally adding, deleting, and modifying database records.

Site logoff is straightforward. Cookies must specify an expiration time. Each time login is tested and a cookie is found, its expiration time is set at $DeadTimeOut seconds in the future. To logoff, the expiration time is simply set to the past causing the cookie to disappear.

"Create account" opens a new browser window (via JavaScript) where AddAcct.pl is run. This program renders a form with input fields for the student's user ID, first, middle, and last names, password and e-mail address. When the "Submit" button is clicked, the passwords database is queried to determine if the user account exists. If the user ID already exists, then the program calls itself, redisplaying the completed form and message that informs the student that the user ID is not available. If the user ID doesn't exist, the information is added to the database as a new record. The program then calls itself, redisplays the form and its data, and a message indicating success.

"Delete account" opens a new browser window (via JavaScript) where the student enters a user ID and password. The database is queried for a record that contains these data. If a record is found, it is deleted, and success is reported. If no record is found, the student is so informed, and the new window is closed effectively passing control back to the menu.

Account editing requires three steps. First, a new window is opened by JavaScript and the user must provide a user ID and password. Next, the password database is queried to validate the user ID and password. Finally, the account information is presented to the user for editing. The user ID cannot be edited because it must be unique. Allowing the user to change this increases the chance of conflicts over a specific user ID. When editing is complete, clicking the submit button saves the data and updates the login cookie expiration.

The final menu choice permits recovery of a forgotten password. The program opens a new window where the user ID and email address are entered. If these two items match the information stored in the passwords database, then the user's password from the database is emailed to the address. The program then re-executes and alerts the user that the password will soon arrive via email. When the message is released, the window closes.

The programs separate program logic (in the main program) from HTML generation (in subprograms). This division effectively separates programming from instructional design to simplify development. Instructional design is implicit in the presented document and consists of the assignment, the instructions, and response entry fields. These components are conveyed via HTML. HTML editors such as Microsoft FrontPage (http://www.microsoft.com/), or Macromedia Dreamweaver (http://www.macromedia.com/software/dreamweaver/), or Macromedia HomeSite (http://www.macromedia.com/software/homesite/) make composing these documents as easy as composing traditional documents with a word processor. Assignments are developed by composing the document with a HTML editor, then cutting and pasting the HTML between the Perl "print <<END_HTML;" command and the "END_HTML" marker in a subroutine.

Program logic is best developed and tested on a local website hosted by Microsoft's Personal Web Server (PWS). Local hosting means that a computer works like a web server for browser requests originating from that computer. This allows the developer to create a model website on a local computer. When fully developed, copying the programs into corresponding directories on a web server deploys them. The programs then work on the World Wide Web server exactly as they did on the local host.

Table 1 shows the directory structure and user permissions for the RoboTA web site. This structure places files that require common permissions and perform common functions the same directory. As a result, files inherit permissions from their directory. Files in the Datafiles directory require read and write permissions for Internet users so that databases can be read and edited. Perl programs require execute permissions to run on the server. The wwwroot directory contains pure HTML files that require only read permissions. The programs that are specific to homework are placed in the Homework directory. Permissions for this directory are inherited from its parent. JavaScript programs run on the client computer so that read permissions are adequate for these files.

As a final consideration, the program developer will need reference materials. Useful references include Powell (HTML), Flanagan (JavaScript), Metzler, and Michalski, or Guehlich, et. al., (CGI), Schwartz and Phoenix, then Wall, Christiansen, and Orwant, (Perl), Roth (Perl Win 32 ODBC extensions) and Dubois (Perl Win32 OLE extensions).

Evaluation

So far attention thus far has focused on how to implement online homework. The normative issue of whether online homework should be implemented will next be examined by considering costs and benefits, and changes in student attitudes and learning performance associated with online homework.

The primary cost-benefit calculation for online homework compares increased homework development time to reduced homework grading time for online as opposed to traditional homework. Online homework requires more development time incurred either directly by an instructor or indirectly by an instructor/programmer team. This time has an opportunity cost. This report and the code available on the accompanying web site should reduce the development time required for online homework but it remains substantial. Also, the development time required for subsequent online assignments is far less than for the first assignment because the program developed for the first assignment can be modified for use in other assignments. However, the development time requirement for any online homework assignment will likely exceed that for a corresponding traditional homework assignment.

The primary benefit of online homework is the practical elimination homework grading time. For traditional homework, grading time is directly related to class size, the number of assignments, the length of the assignment, and the type of the assignment. Therefore, courses with large enrollments, numerous homework assignments with many problems, and homework assignments that have numerical answers resulting from the application of complex formulae are more likely to display positive net instructional-time benefits from utilizing online homework. Online homework is capital-like in that benefits accrue during subsequent offering of the course and to other courses where online homework is utilized. As a result, the cost-benefit calculations of instructional time should not be course specific but should take into account all offerings of all courses where it will be used.

Other benefits might be due to changes in instructional design made possible by online homework. For example, online homework might reduce the instructional time required to teach a course, *ceteris paribus*. This time saved might be devoted to additional homework assignments. If additional homework creates additional learning and if online homework makes more homework assignments possible, then a *ceteris paribus* cost-benefits measure of instruction-time will understate online homework's benefits. Table 2 shows data related to this issue from the author's fall 2001 economics of futures markets course. This is an upper-division, undergraduate, large lecture course. Seven online homework assignments were used in the 2001 fall semester but grading time requirements would have dictated the use of fewer traditional assignments. Eleven examination questions were directly related to the homework assignments. Between 105 and 111 students took the three course examinations. Table 2 indicates that higher homework scores are associated with greater likelihood of correctly responding to the corresponding examination item. This relationship is statistically significant (a chi square value of 13.45, the probability of a greater chi square with 2 degrees of freedom is 0.0012). Hence, additional homework assignments made possible by the online technology create additional learning that is not recognized in instructional-time cost benefit computations.

Online homework might create other learning benefits due to its convenience, interactivity, or other differentiating characteristics. This possibility was investigated by surveying the students in the author's economics of futures markets course. Students were asked to indicate the usefulness of homework problems, the course web site, and lectures in helping them learn the course material for the 1997, 2000, and 2001 course offerings. Homework assignments were not on the Internet in 1997 but in 2000 and 2001 they were. The online approach was developed in the intervening years. These data are summarized in table 3.

Table 3 presents a comparison of the usefulness of online homework (2000 and 2001) to traditional homework (1997), which shows a three percent increase in the relative frequency of the very useful response, and a three percent decrease in the somewhat useful response, and no change in the not useful response. Thus, students were slightly more favorable toward online homework than toward traditional homework, but the difference is not significant.[5] In short, these results indicate that the medium doesn't matter so that instructional time savings and the potential for administering additional homework assignments should serve as the primary evaluative criteria for online homework.

Online homework alters a course web site and potentially alters student attitudes regarding the usefulness of the web site. The data in table 3 indicate that the development of online homework was associated with a significant increase in the perceived usefulness of the course web site. Much of this increase results as online homework requires students to use the web site, and hence to develop an opinion on the usefulness of the site. In 1997, 35 respondents indicated that they didn't use the course web site. This group disappeared in 2000 and 2001. Table 3 compares the attitudes of the website users and indicates that the perceived usefulness of the web site increased significantly among the site users.[6] From this we conclude that even though the homework medium doesn't seem to matter, the integration of homework into a course web site increases the perceptions of the usefulness of the entire site.

5 The chi-square statistic for the equality of relative frequencies in the two rows is 0.193, has two degrees of freedom, and a probability of a greater value of 90.8%. Hence, the null hypothesis of no difference in the relative frequencies cannot be rejected.

6 For only web site users, the computed chi-square statistic is 21.7 with 2 degrees of freedom. The probability of a larger value is less than 0.0001.

The final item in table 3 addresses concerns about course web sites displacing classroom lectures. These data show increased student perceptions of the usefulness of lectures. Though positive this effect is not significant.[7] While the improvement in student attitudes is welcome, the lack of significance is not surprising as the lecture content and presentation methods were largely constant for the classes evaluated.

Summary and Conclusions

This paper presents programs to implement online homework on a course web site. These programs are written in Perl, a freely available and easy to use scripting language. Other instructors can use these programs to reduce the costs of adopting online homework. The programming techniques for shown can be adapted for other interactive Internet-based instructional applications such as surveying, testing, creating simulations and reporting grades.

Our second objective was to evaluate online homework based on its use in an economics course. The results indicate that online homework offers no intrinsic advantage. Online homework was viewed as slightly more useful than traditional homework, but not insignificantly so. This lead to the conclusion that the primary criteria for evaluating online homework compared to traditional homework is the potential decrease in grading time versus the increase in development time. As a general rule, online homework has the greatest advantage when development costs can be spread over large numbers of students in initial, subsequent and derived applications. When online homework saves instructional time, the time saved can be reinvested in new instructional devices, which can lead to increased learning.

[7] The computed chi-square statistic is 4.11 with two degrees of freedom. The probability of a greater value is 0.128.

The adoption of online homework can create positive externalities for other aspects of a course. Student perceptions of the usefulness of the course web site increased with the development of online homework. Two factors account for this. First, online homework requires students to use the web site so the usefulness of the web site increases by definition. However, among only the web site users, the perception of web site usefulness also increased. This was likely due to the online homework programming techniques being used in other applications that were added to a course web site. These course web site developments made it more useful.

Finally, concerns about online instruction replacing traditional lectures appear to be misplaced. An increase in the perceived usefulness of lectures accompanied the development of the course web site. We thus conclude that online homework is a course web site development that offers economic benefits to the instructor with no learning sacrifice for students. Improved student attitudes toward lectures indicates that the lectures and web sites are more like complements than substitutes for teaching economics.

Table 1. Directories and files required.

Folders and files	Privileges	Notes
/Inetpub		Root directory for IIS and PWS content
/RoboTA		Directory for this application
/Datafiles	**Read, Write**	
RoboTA.mdb		MS Access database. Connected by ODBC with ODBC DSN = RoboTA
Passwords		Table: names, passwords, etc of registered users
HW1data		Table: data for HW1
HW1scores		Table: recorded scores for HW1
/Homework		
HW2.dat		Log of user operation for homework 2
/Gradebook		
/Scripts	**Read, Execute**	
AddAcct.pl		Perl prog to add user account
DelAcct.pl		Perl prog to delete user account
GetPW.pl		Perl prog to retrieve password
EditAcct.pl		Perl prog to edit account info (name, password, email address, etc.)
Logoff.pl		
Logon.pl		
OLEQuery.pl		Perl subprograms used by the above
/Homework		
HWgate.cgi		Calls scripts HW1, HW2, and etc. pass through here
HW1edit.cgi		
HW2.pl		
Etc.		
/Gradebook		
/wwwroot	**Read**	Directory containing all HTML files
HWMenu.htm		Menu for homework problems
/images		jpg, bmp, and other image files
/javascript		JavaScript programs included in html files
cookies.js		
validate.js		Input validation program
/Perl		Directory where Perl is installed.

Table 2. Homework score versus response on related examination items, fall semester 2001.[a]

	Score on related homework assignment[b]			
Exam response:	0%	Partial credit	100%	Total
Incorrect	45%	42%	33%	38%
Correct	55%	58%	66%	62%
Count	245	412	517	1174

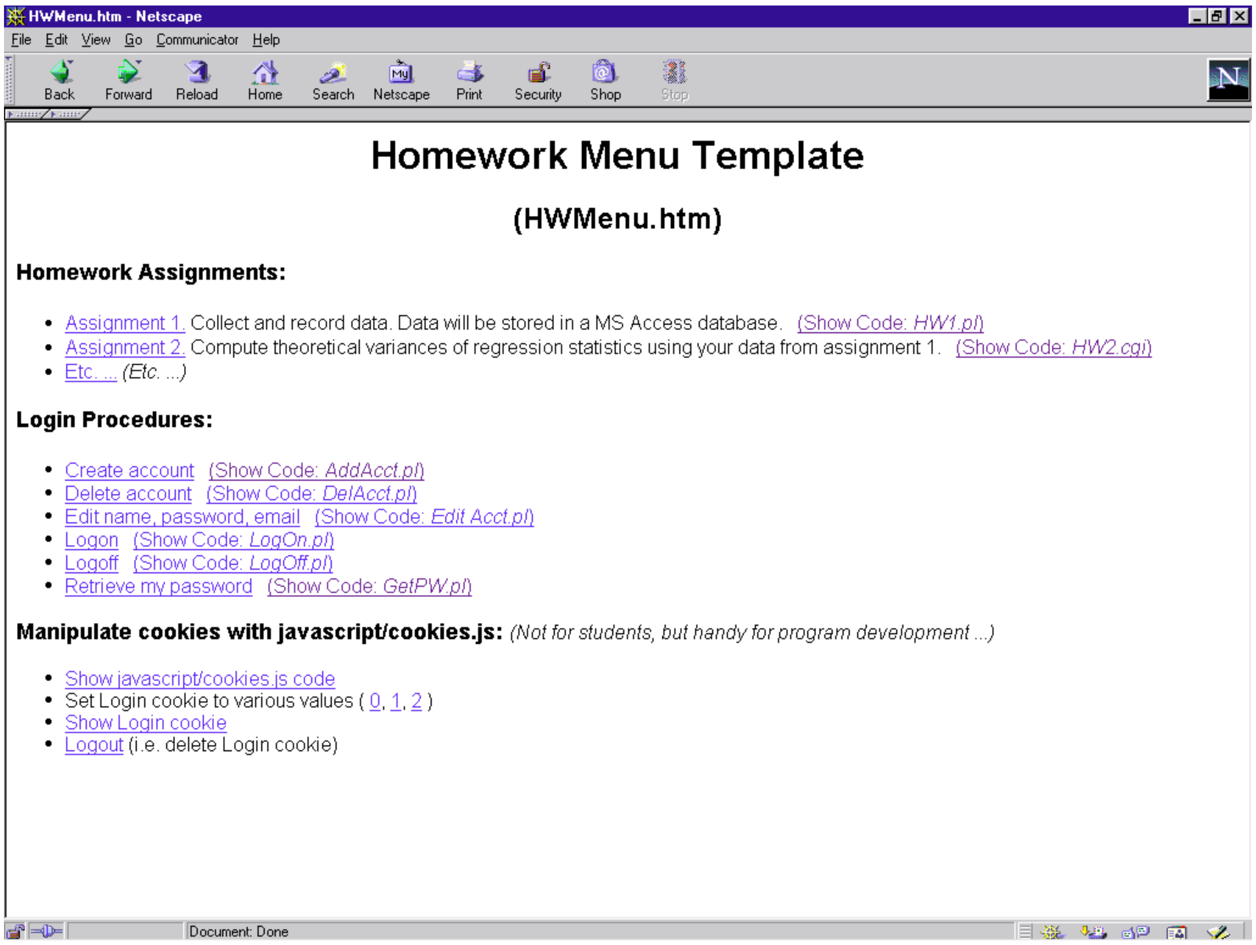
a/ Seven homework assignments, eleven related examination questions on three different examinations, 105 to 111 student (depending on the examination).

b/ Percentages are of column counts.

Table 3. Relative frequency of responses to survey item, "Please indicate the value of these mechanisms in helping you learn about futures markets."

Mechanism	Response and relative frequency[a]			
Year	Very useful	Somewhat useful	Not useful	Response[a] Count
Homework problems				
1997	43.5%	44.9%	11.6%	69
2000&01	46.5%	41.9%	11.6%	155
Course website				
1997	33.3%	45.5%	21.2%	33
2000&01	72.4%	23.1%	4.5%	156
Lectures				
1997	44.3%	41.4%	14.3%	70
2000&01	58.7%	31.6%	9.7%	155

a/ Number of respondents: 73, 68, and 88 in 1997, 2000, and 2001 respectively.


HWMenu.htm - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Document: Done

Homework Menu Template

(HWMenu.htm)

Homework Assignments:

- Assignment 1. Collect and record data. Data will be stored in a MS Access database. (Show Code: *HW1.pl*)
- Assignment 2. Compute theoretical variances of regression statistics using your data from assignment 1. (Show Code: *HW2.cgi*)
- Etc. ... *(Etc. ...)*

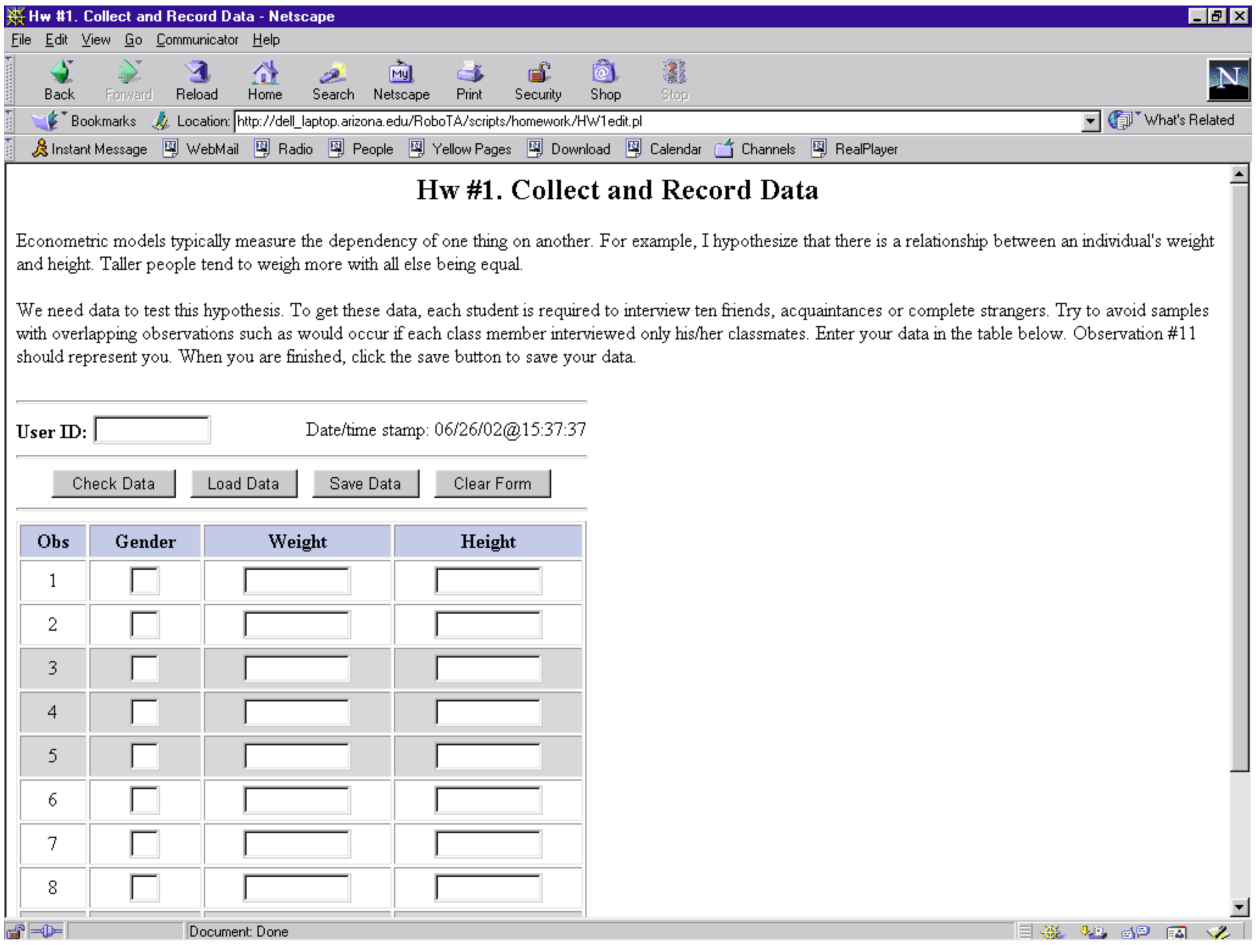
Login Procedures:

- Create account (Show Code: *AddAcct.pl*)
- Delete account (Show Code: *DelAcct.pl*)
- Edit name, password, email (Show Code: *Edit Acct.pl*)
- Logon (Show Code: *LogOn.pl*)
- Logoff (Show Code: *LogOff.pl*)
- Retrieve my password (Show Code: *GetPW.pl*)

Manipulate cookies with javascript/cookies.js: *(Not for students, but handy for program development ...)*

- Show javascript/cookies.js code
- Set Login cookie to various values (0, 1, 2)
- Show Login cookie
- Logout (i.e. delete Login cookie)

Figure 1. Homework menu screen.


Hw #1. Collect and Record Data - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Location: http://dell_laptop.arizona.edu/RoboTA/scripts/homework/HW1edit.pl
What's Related
Instant Message WebMail Radio People Yellow Pages Download Calendar Channels RealPlayer
Document: Done

Hw #1. Collect and Record Data

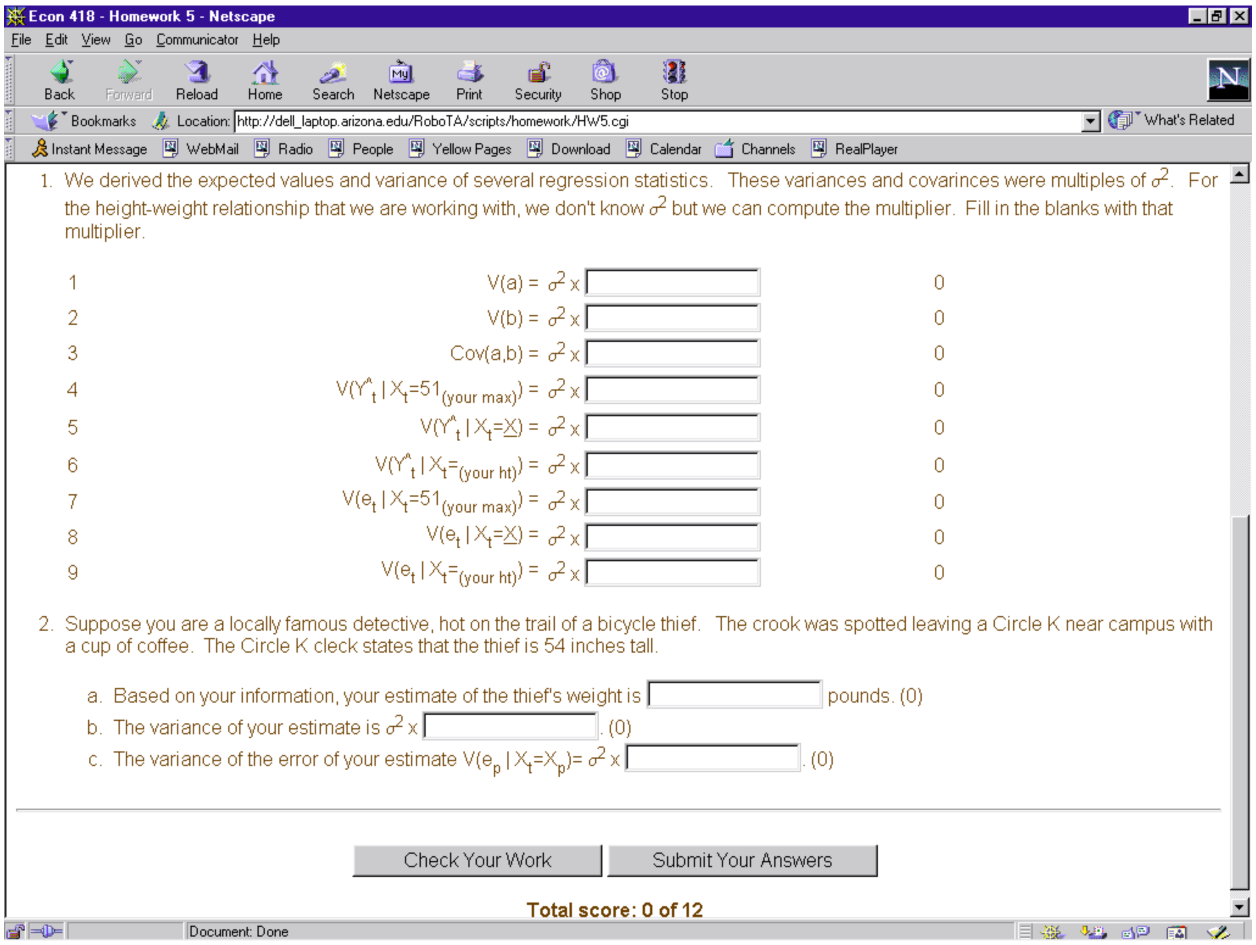
Econometric models typically measure the dependency of one thing on another. For example, I hypothesize that there is a relationship between an individual's weight and height. Taller people tend to weigh more with all else being equal.

We need data to test this hypothesis. To get these data, each student is required to interview ten friends, acquaintances or complete strangers. Try to avoid samples with overlapping observations such as would occur if each class member interviewed only his/her classmates. Enter your data in the table below. Observation #11 should represent you. When you are finished, click the save button to save your data.

User ID: Date/time stamp: 06/26/02@15:37:37

Obs	Gender	Weight	Height
1			
2			
3			
4			
5			
6			
7			
8			

Figure 2. Homework assignment 1 – collect and record data.



Econ 418 - Homework 5 - Netscape
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Shop Stop
Bookmarks Location: http://dell_laptop.arizona.edu/RoboTA/scripts/homework/HW5.cgi
What's Related
Instant Message WebMail Radio People Yellow Pages Download Calendar Channels RealPlayer

1. We derived the expected values and variance of several regression statistics. These variances and covarinces were multiples of σ^2. For the height-weight relationship that we are working with, we don't know σ^2 but we can compute the multiplier. Fill in the blanks with that multiplier.

1	$V(a) = \sigma^2 \times$ ______	0
2	$V(b) = \sigma^2 \times$ ______	0
3	$Cov(a,b) = \sigma^2 \times$ ______	0
4	$V(\hat{Y}_t \mid X_t=51_{(your\ max)}) = \sigma^2 \times$ ______	0
5	$V(\hat{Y}_t \mid X_t=\underline{X}) = \sigma^2 \times$ ______	0
6	$V(\hat{Y}_t \mid X_t=_{(your\ ht)}) = \sigma^2 \times$ ______	0
7	$V(e_t \mid X_t=51_{(your\ max)}) = \sigma^2 \times$ ______	0
8	$V(e_t \mid X_t=\underline{X}) = \sigma^2 \times$ ______	0
9	$V(e_t \mid X_t=_{(your\ ht)}) = \sigma^2 \times$ ______	0

2. Suppose you are a locally famous detective, hot on the trail of a bicycle thief. The crook was spotted leaving a Circle K near campus with a cup of coffee. The Circle K cleck states that the thief is 54 inches tall.

   a. Based on your information, your estimate of the thief's weight is ______ pounds. (0)
   b. The variance of your estimate is $\sigma^2 \times$ ______. (0)
   c. The variance of the error of your estimate $V(e_p \mid X_t=X_p)= \sigma^2 \times$ ______. (0)

Total score: 0 of 12

Figure 3. Homework assignment 2 – compute variance of regression statistics.

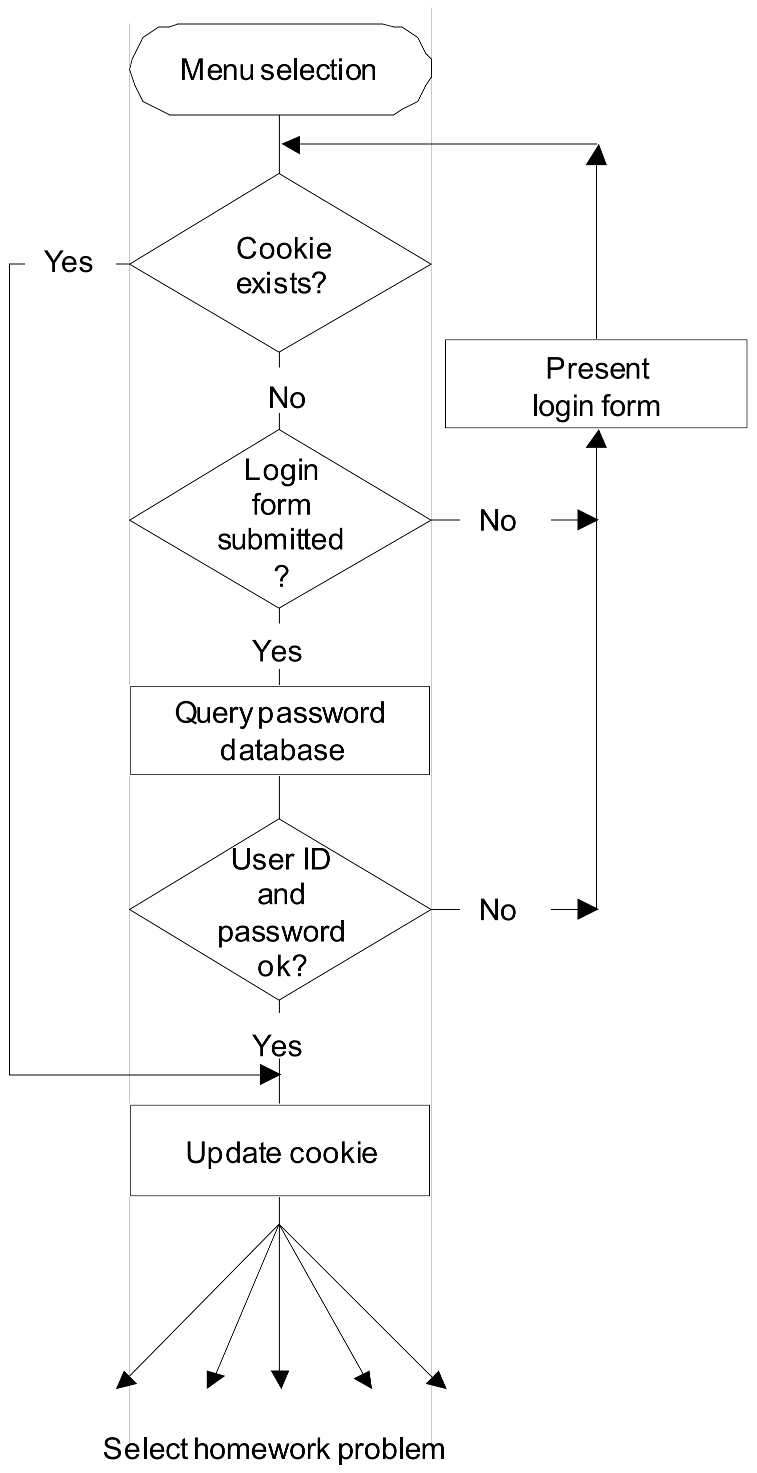


Figure 4. Login procedure.

References

Agarwal, R. and A.E. Day. 1998. The impact of the Internet on economic education. Jour. of Economic Education 29(2): 99-110.

Becker, W.E. 1997. Teaching economics to undergraduates. Jour. of Economic Literature 35(3): 1347-1373.

Bloom, B.S., M.D. Engelhart, E.J. Furst, W.H. Hill, and D.R. Krathwohl. *Taxonomy of Educational Objectives, Handbook 1: Cognitive Domain.* ed. B.S. Bloom. New York: David McKay Company, Inc. 1956.

Dahlgran, R.A. 2002. Internet Usage in Agricultural Economics Instruction. NACTA Journal (forthcoming).

Dubois, J. 1998. "Win32::OLE." The Perl Journal. Issue #10, Vol. 3 No. 2 (Summer). Available online at http://www.samag.com/documents/s=1286/sam03020008/.

Guelich, S., S. Gundavaram, and G. Birznieks. 2000. CGI Programming with Perl, 2nd Edition. O'Reilly & Associates, Inc., Sebastapol, CA.

Meltzer, K., and B. Michalski. 2001. Writing CGI Applications with Perl. Addison-Wesley.

Mercer. D. 2001. ASP 3.0 A Beginners Guide. Osborne/McGraw-Hill. Berkeley, CA.

Buser, D., J. Kauffman, J.T. Llibre, B. Francis, D. Sussman, C. Ullman, J. Duckett. 1999. Beginning Active Server Pages 3.0. Wrox Press, Birmingham, UK.

Powell, T.A. 2001. HTML: The Complete Reference, Third Edition. Osborne/McGraw-Hill, Berkeley, CA.

Ramstad, E. 2001. A campus connected. The Wall Street Jour. Mar 12: R25.

Reaves, W., A. McGuirk, and H. Deskins. "Assessments of the Educational Value of Web-Based Instructional Materials for Introductory Agricultural Economics." Review of Agricultural Economics, 23(2001): 492-509.

Reselman, B. 2000. Active Server Pages 3.0. Que Corporation, Indianapolis, IN.

Roth, D. 2002. Win 32 Perl Programming: The Standard Extensions, 2nd Edition. New Riders Publishing, (www.newriders.com), Indianapolis, IN.

Schwartz, R.L. and T. Phoenix. 2001. Learning Perl, 3rd Edition. O'Reilly & Associates, Inc., Sebastapol, CA.

Vachris, M.A. 1999. Teaching principles of economics without 'chalk and talk': The Experience of CNU Online. Jour. of Economic Education 30(3): 292-303.

Wall, L., T. Christiansen, and J. Orwant. 2000. Programming Perl, 3rd Edition. O'Reilly & Associates, Inc., Sebastapol, CA.