2011

# Technical Note: Detecting and Subcategorizing Hard-coding Errors in Bioenergy-relevant Spreadsheets Using Visual Basic for Applications (VBA)

Vertika Rawat
*Iowa State University*

D. Raj Raman
*Iowa State University*, rajraman@iastate.edu

Robert P. Anex
*University of Wisconsin–Madison*

# Technical Note: Detecting and Subcategorizing Hard-coding Errors in Bioenergy-relevant Spreadsheets Using Visual Basic for Applications (VBA)

**Abstract**
Electronic spreadsheets play an indispensable role in the simulation, modeling, and analysis of bioenergy systems, and their results have the ability to affect decision-making significantly. Prior research has shown that spreadsheets are highly error-prone, and that a large percentage of these errors are difficult to detect. To that end, we developed computer code (implemented in Visual Basic for Applications, running under Microsoft Excel) to detect a particularly insidious form of spreadsheet error: the hard-coding error. These errors are defined as the presence of one or more unreferenced numerical values in a cell formula. The code was used to audit six engineering spreadsheets relevant to bioenergy systems, three developed in our lab (and reported on in other sessions at the AIM), and three in the public domain. The preliminary audit results were analyzed to understand the nature and distribution of hard-coding errors. The preponderance and diversity of hard-coding errors in these spreadsheets motivated us to subcategorize them. Together, the hard-coding error detection program and sub-categorization program provide a robust and rapid means of detecting and categorizing multiple types of hard-coding errors. Use of these programs could increase the reliability of spreadsheet software used in simulation, modeling, and analysis of bioenergy systems.

**Keywords**
Hard-coding errors, Bioenergy, Decision-making, Error detection, Subcategorize, Visual Basic for Applications

**Disciplines**
Agriculture | Bioresource and Agricultural Engineering

**Comments**
This article is from *Applied Engineering in Agriculture*, 27, no. 3 (2011): 469–474.

Technical Note:

# Detecting and Subcategorizing Hard-coding Errors in Bioenergy-relevant Spreadsheets Using Visual Basic for Applications (VBA)

V. Rawat, D. R. Raman, R. P. Anex

**Abstract.** *Electronic spreadsheets play an indispensable role in the simulation, modeling, and analysis of bioenergy systems, and their results have the ability to affect decision-making significantly. Prior research has shown that spreadsheets are highly error-prone, and that a large percentage of these errors are difficult to detect. To that end, we developed computer code (implemented in Visual Basic for Applications, running under Microsoft Excel) to detect a particularly insidious form of spreadsheet error: the hard-coding error. These errors are defined as the presence of one or more unreferenced numerical values in a cell formula. The code was used to audit six engineering spreadsheets relevant to bioenergy systems, three developed in our lab (and reported on in other sessions at the AIM), and three in the public domain. The preliminary audit results were analyzed to understand the nature and distribution of hard-coding errors. The preponderance and diversity of hard-coding errors in these spreadsheets motivated us to subcategorize them. Together, the hard-coding error detection program and sub-categorization program provide a robust and rapid means of detecting and categorizing multiple types of hard-coding errors. Use of these programs could increase the reliability of spreadsheet software used in simulation, modeling, and analysis of bioenergy systems.*

***Keywords.*** *Hard-coding errors, Bioenergy, Decision-making, Error detection, Subcategorize, Visual Basic for Applications.*

The versatility of spreadsheets has led to their extensive application at all levels of organizations. Because of their wide use, concerns have been raised about the integrity and validity of spreadsheets, as stated by Galletta et al. (1997), and many other authors including Powell et al. (2009) have shown that spreadsheets are highly vulnerable to errors. Users cannot readily detect the majority of such errors, which could result in potentially devastating miscalculations in many settings. The typical approach to debugging spreadsheets involves doing hand calculations to verify the results – unfortunately, this approach is time consuming and is frequently skipped or done cursorily. Furthermore, even if the spreadsheet is providing correct results with one set of input data, hidden errors can mean that when inputs change, incorrect values result.

With the pervasiveness of spreadsheet use, they are increasingly being used for mission-critical applications. Consequently, errors in spreadsheets can lead to making sub-optimal decisions as discussed by Teo and Lee-Partridge (2001). These errors cost the organizations that rely on them millions of dollars (EUSPRIG, n.d.). Panko (1999) showed that human-based code inspection – either in groups or individually – was only 60% to 80% effective at capturing errors in spreadsheets. Panko did not estimate the cost of such inspections, which would likely show human error detection is extremely expensive. A systematic and automated method of error detection could serve to reduce error rates and make spreadsheets more reliable.

A first step in developing any type of automated error detection system is to characterize the types of errors that can occur. To this end, Rajalingham et al. (2008) proposed an elaborate taxonomy for spreadsheet errors, wherein errors are broadly categorized as system-generated or user-generated. User-generated errors are further decomposed into qualitative or quantitative errors. Quantitative errors are numerical errors that lead to incorrect bottom-line values, as opposed to qualitative errors, which do not immediately produce incorrect numeric values but degrade the quality of the model.

Quantitative errors are further subdivided into accidental errors (due to typing errors), omission errors (failure to consider one or more important parameters), alteration errors (making changes to the model) and duplication errors (re-creating elements of the model). They could also fall into the categories of domain knowledge errors (stemming from a lack of knowledge), mathematical representation errors (due to inaccurate construction of a formula), or logic/syntax errors (due to erroneous logic or syntax).

Qualitative errors are trifurcated into structural errors (resulting from flaws in the design or lay-out of the model),

Submitted for review in September 2010 as manuscript number IET 8762; approved for publication as a Technical Note by the Information & Electrical Technologies Division of ASABE in March 2011. Presented at the 2010 ASABE Annual Meeting as Paper No. 1009201.

The authors are **Vertika Rawat, ASABE Member Engineer,** Graduate Student, **Dave Raj Raman,** Assistant Professor, Department of Agricultural and Biosystems Engineering, Iowa State University, Ames, Iowa; **Robert P. Anex, ASABE Member Engineer,** Professor, Department of Biological Systems Engineering, University of Wisconsin-Madison, Madison, Wisconsin. **Corresponding author:** Dave Raj Raman, Department of Agricultural and Biosystems Engineering, Iowa State University, 3222 NSRIC, Ames, IA 50011-3310; phone: 515-294-0465; e-mail: rajraman@iastate.edu.

temporal errors (from the use of data which has not been updated), and maintainability errors (from spreadsheet features which make it difficult to be modified). An extremely common maintainability error is the hard-coding error. Hard-coding errors (HCE) are defined in the literature as the use of raw numerical value(s) in cell formulae. For example, "=A3*2.204" or "=C7/365" are both HCE, whereas "2.204" or "365" or "1" coded into a cell are not, because the numerical value is not embedded in a formula. It is noteworthy that formula cells have a disproportionately high share of errors, e.g., Powell et al. (2008) stated that approximately 80% of errors documented by European Spreadsheet Risks Interest Group (EUSPRIG) occurred in formula cells. The term hard-coded applies because it renders the formula, and hence the whole spreadsheet, inflexible to changing values in future scenarios. Updating a model containing HCE is time consuming because of the dispersion of numerical data throughout the spreadsheet.

Powell et al. (2009) applied a spreadsheet auditing protocol to 50 diverse operational spreadsheets, and reported that hard-coding errors were the most common (43.5% of erroneous cells), followed by logic errors (28.6% of erroneous cells) and reference errors (22.1% of erroneous cells). The remaining categories in their own interim error taxonomy including copy/paste, omission, and data input errors together accounted for less than 5% of erroneous cells. In addition to their high frequencies of occurrence, hard-coding errors are cumbersome to detect manually.

However, hard-coding is vulnerable to automated detection, and in this article we report on the results of a spreadsheet auditing effort in which hard-coding errors were automatically identified and subcategorized, thus addressing a need for such information identified by prior workers (e.g., Powell et al., 2008). An attempt to find errors using multiple manual strategies was made by Galletta et al. (1997), but did not prove to be very effective. We have tried to take a step forward in that direction and have developed programs for hard-coding error detection in-house. As we scrutinized our audit results, we realized the importance of subcategorizing hard-coding errors when dealing with engineering spreadsheets, and added a second program with subcategorization capabilities. These capabilities extend beyond what is available commercially (e.g., auditing and error-checking tools available in Excel, and from third-party firms such as XL Analyst, http://www.codematic.net, and Spreadsheet Professional, http://www.spreadsheetinno vations.com/). We then applied the pair of programs to multiple bioenergy-relevant spreadsheets.

## Materials and Methods

### Overview

Both programs were written in Microsoft Excel Visual Basic for Applications (VBA). The first program identified hard-coding errors and presented a summary of error statistics and a detailed error report on a new worksheet tab. This tab was labeled HCER (Hard-Coding Error Report). The first program also flagged error cells in the respective worksheets using shading and font bolding to make it easy for users to locate them. The second program scanned the HCER summary, and subcategorized the errors into four unique types.

### Algorithms

The first program (namely "HCD" – Hard-Coding Detector) stores all worksheet names in the workbook in a string array. Worksheets that are strictly charts/graphs are automatically skipped. The program displays the worksheet count and queries the user to see if there are any protected sheets in the workbook. If there are any protected worksheets, the cell shading and bolding functions are disabled. Because the detection algorithm can be misled by worksheet names containing numbers (e.g., "TAB_44"), the user is prompted to enter new names for any such worksheets, and the program assigns the new names. The program uses built-in functions to find row and column bounds of data for each worksheet, thus greatly reducing runtime. On each worksheet, the program loops through all cells within the data bounds. Once a formula cell is found, the formula is stored in a string and parsed. If a number is encountered as the string is parsed, a check is made on the preceding element. If the predecessor turns out to be a letter, the program assumes that a cell address is specified, not an unreferenced numerical value. If this is the case, the program checks the successor string element too, skipping the successive string elements, as long as they are numbers. However, if the predecessor to a number was not a letter, a hard-coding error is flagged. A counter variable keeps track of the number of such instances. If a hard-coding error has been detected, the numerical value is checked to see if it is equal to one. If this numeral is "1" there is another counter variable that keeps track of the number of unity occurrences. The loop continues until the last element of the formula string of the concerned cell. At the end of this, the two counter variables are compared. If they are equal, the cell is solely suffering from the "unity" error.

The second program (namely "SubCat") subcategorizes the hard-coding errors detected by the first program, by reading the formula of the faulty cell into a string, and then parsing it. When it runs into a number, it employs the same strategy described earlier to distinguish valid cell addresses from unreferenced numerical values. When the program locates an unreferenced numerical value, it subcategorizes according to the taxonomy shown in table 1. If a cell contains multiple subcategories, each type is captured and reported. While HCE already exist in the taxonomies on spreadsheet errors (e.g., Rajalingham et al., 2008; Powell et al., 2009), this is the first time they are being subcategorized to our knowledge.

### Interface

A series of dialog boxes are used for the primary user interface for the first program (the second program does not require any such dialog boxes). Message boxes, input boxes, and radio buttons are used as follows: (a) to display the total number of worksheets in the workbook; (b) to respond to whether there are any protected worksheets in the workbook; (c) to display the tab names of worksheets which contain number(s); (d) to enter the new tab names for worksheets with numbers (e) to choose the background color and font of the cells to be flagged.

### Error Statistics/Output

After HCD is finished running on all the selected worksheets, it displays the total number of cells checked, the number of cells with hard-coding errors, and the correspond-

**Table 1. Taxonomy of hard coding errors implemented by program, indicating type, description, example, and comments.**

| Error Type | Description | Example | Comments |
|---|---|---|---|
| Unity errors | The presence of the value 1 as an unreferenced numerical value in the cell formula | B6 = (1-A5)/B14 | The first "1" is the error, the second occurrence is not flagged because it is part of the cell address |
| Power of 10 | The presence of numbers like 10, 100, 1000, and so on as an unreferenced value | C21 = (B10-A2)*100 | The "100" is the error, the "10" in B10 is not flagged because it is part of the cell address |
| Commonly used unit conversions | The presence of common unit conversion factors relevant to bioenergy | D4 = (C4*3.785)/(B2-D4) | The 3.785 is most likely a gallon-to-liter conversion |
| Other unidentified numerals | The presence of numerical values other than unity, power of 10, and unit conversion factors, as unreferenced values | G4 = (D4-13.9)*(8 + G2) | Both the "8" and the "13.9" are errors of this type |

ing cell error rate (CER) of the audited workbook, in a popup box. The Cell Error Rate (CER), a generic term coined by Panko and Halverson (1996), refers to the frequency of error cells as a percentage of total cells in consideration. The complete error statistics also including the number of cells with hard-coding errors that are uniquely unity errors can be viewed on the Hard-Coding Error Report tab (denoted by "HCER"). The HCER worksheet is created by the program after the last used worksheet of the workbook.

To facilitate rapid review of all errors, the HCER also presents a list of each error detected, sorted by worksheet, indicating both cell reference and the equation in the cell. If the only hard-coding error in a cell is a unity error, it is displayed with a grey fill to be easily distinguished from others. This helps the user to rapidly scan through the HCER overlooking the grey-fill cells suffering from only unity errors (unique), as they tend to be far less dangerous than the others. The unity-error (unique) distinction is made because in certain formulae in engineering spreadsheets – such as when converting dry-basis to wet-basis moisture content – the use of a numerical one is justified and not indicative of a typical hard-coding error, as mentioned by Powell et al. (2008).

SubCat creates a subcategorization table (next to the report generated by the first program) on the same HCER worksheet. The subcategorization statistics include the frequency of (1) unity errors, (2) power of 10 conversions, (3) commonly used unit conversions, and (4) other unidentified numerals. The table shows the number of instances of each of the above type in each faulty cell detected by HCD.

We report the frequencies of errors identified by our programs distributed both across error types and across spreadsheets. To our knowledge, this is the first data on analysis of hard-coding errors and their subcategorization to appear in research literature on spreadsheet errors.

### Audit of Bioenergy-relevant Spreadsheets

The programs were used on the following six diverse workbooks related to simulation, modeling, and analysis of bioenergy systems: The Cob-Cost workbook designed by Carol Faulhaber (MS student, Iowa State University) computes amortized grassroots capital cost of corn-cobs storage systems. The Simple Framework for Analyzing Anaerobic Digestion (S-FAAD) workbook, by Faulhaber and Raj Raman evaluates the economic viability of anaerobic digestion using a set of operating parameters and scale factors. The Framework for the Evaluation of Bioenergy Feedstocks (FEBEF) was developed by Raj Raman and Katrina Christiansen (Ph.D. student, Iowa State University) to provide insight into the relative costs and lifecycle impacts of algae, switchgrass, Miscanthus, and corn. The GREET-BESS Analysis Meta-Model (GBAMM) (Energy and Resources Group, University of California, Berkeley, Calif.) compares life cycle global warming intensity estimates for corn ethanol as computed in BESS (Biofuel Energy Systems Simulator) and GREET (Greenhouse Gases, Regulated Emissions, and Energy Use in Transportation) to understand why the results from the two models are so disparate. The Ethanol-Profitability D1-10 workbook (Ag Decision Maker, Iowa State University Extension, Ames, Iowa), presents an economic model of a typical northern Iowa corn ethanol plant to help track its profitability of corn ethanol production. The GREET Model (Argonne National Laboratory, Argonne, Ill.) is a comprehensive model evaluating the energy use and emissions for diverse scenarios; GREET is available for the research community to use, and has been used in hundreds of refereed journal articles.

## Results and Discussion

### Sample Results

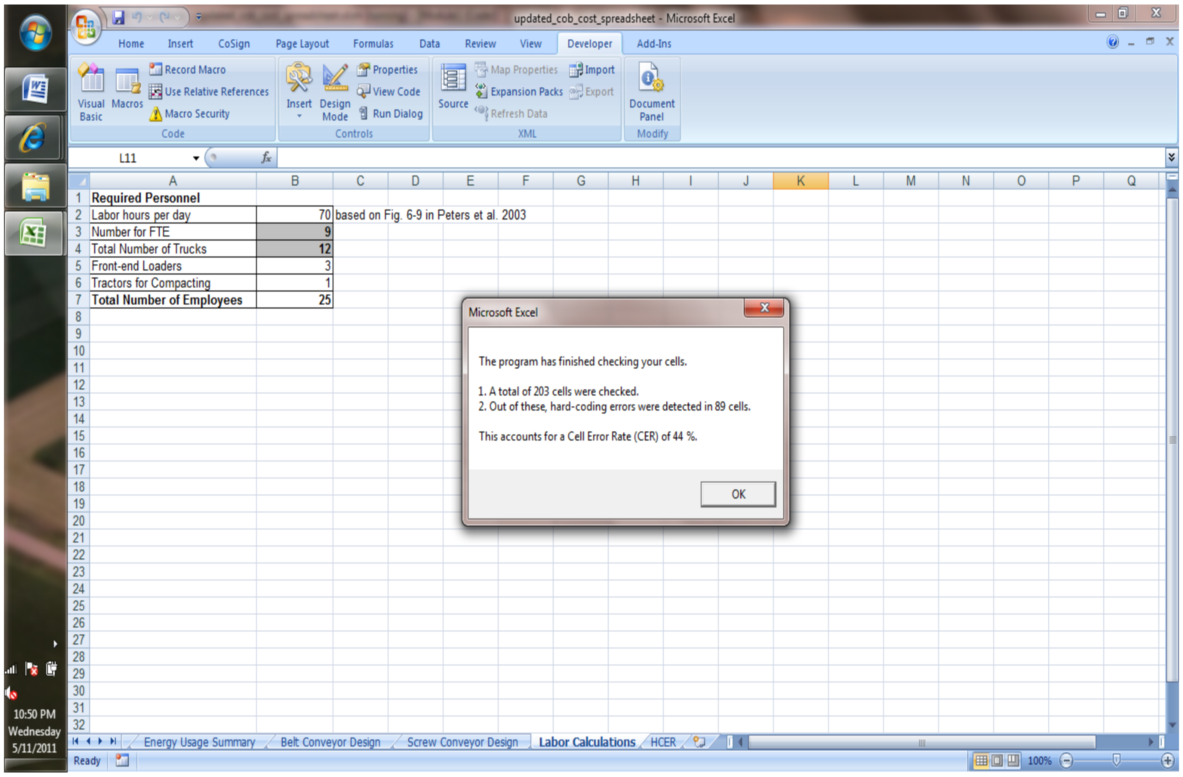
Figures 1 and 2 illustrate output from running HCD on a sample spreadsheet. Figure 3 illustrates the output from running SubCat on a sample spreadsheet.

### Subcategorization of Hard-coding Errors

In light of the large number of hard-coding errors detected in the sample spreadsheets, it appeared useful to further subcategorize them into the following:

- unity errors,
- power of 10 conversions,
- commonly used unit conversion factors,
- other unidentified numerals.

Although power of 10 conversions can be unit conversion factors as well, they form a class of their own and have an overwhelming occurrence rate compared to the other commonly used unit conversion errors. For this reason, we chose to separate them from the other commonly used unit conversion errors.



|  | A | B | C |
|---|---|---|---|
| 1 | **Required Personnel** |  |  |
| 2 | Labor hours per day | 70 | based on Fig. 6-9 in Peters et al. 2003 |
| 3 | Number for FTE | 9 |  |
| 4 | Total Number of Trucks | 12 |  |
| 5 | Front-end Loaders | 3 |  |
| 6 | Tractors for Compacting | 1 |  |
| 7 | **Total Number of Employees** | 25 |  |

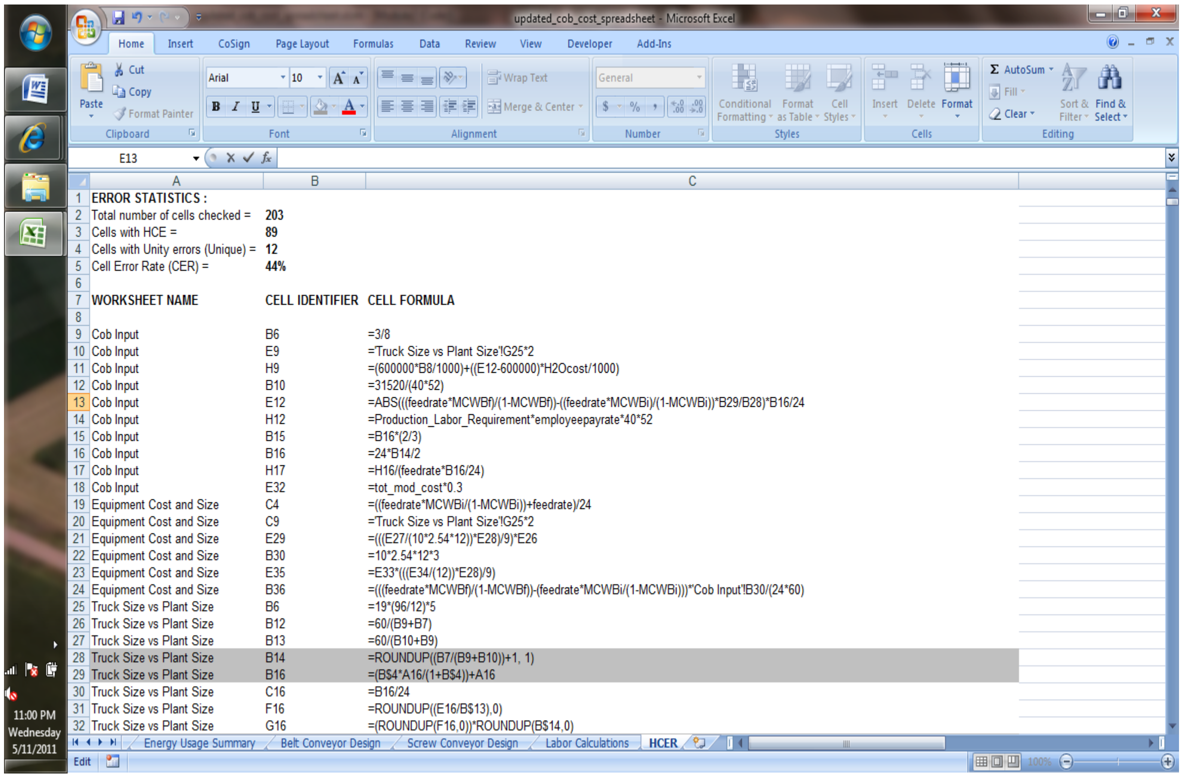
Microsoft Excel

The program has finished checking your cells.

1. A total of 203 cells were checked.
2. Out of these, hard-coding errors were detected in 89 cells.

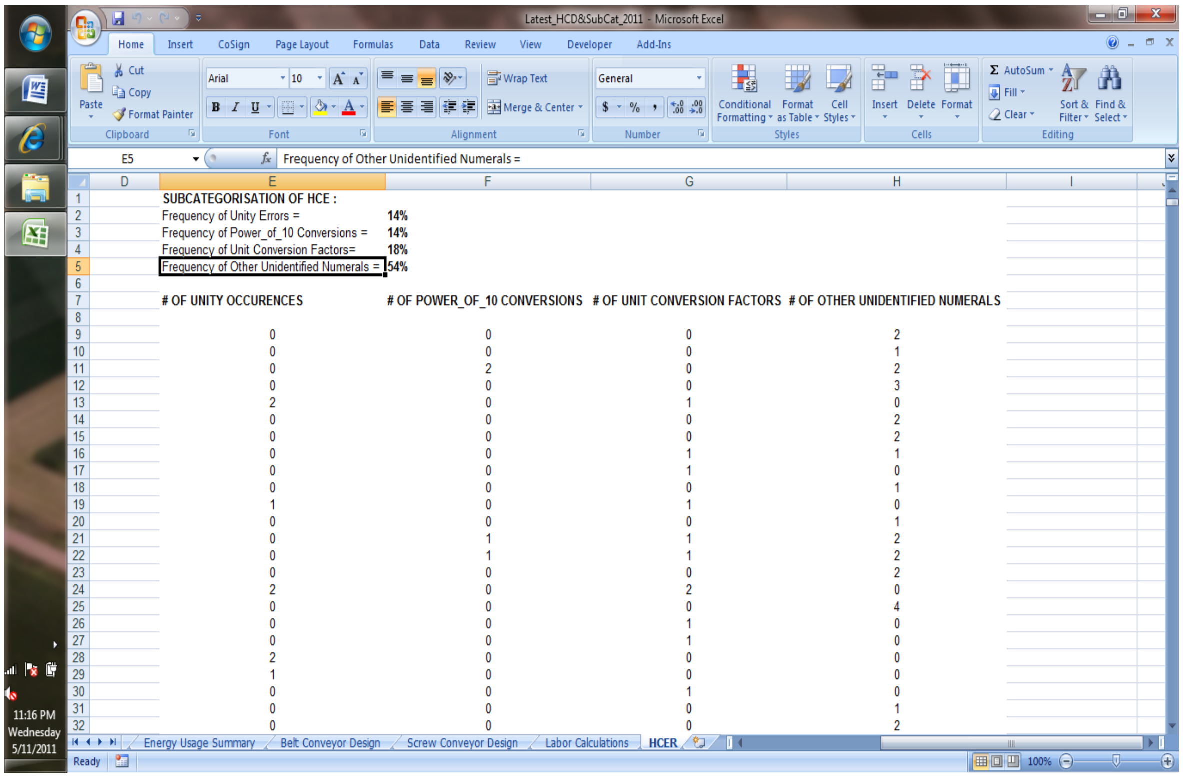
This accounts for a Cell Error Rate (CER) of 44 %.

**Figure 1. Screenshot of the final pop-up message box produced by HCD.**



|  | A | B | C |
|---|---|---|---|
| 1 | ERROR STATISTICS : |  |  |
| 2 | Total number of cells checked = | 203 |  |
| 3 | Cells with HCE = | 89 |  |
| 4 | Cells with Unity errors (Unique) = | 12 |  |
| 5 | Cell Error Rate (CER) = | 44% |  |
| 6 |  |  |  |
| 7 | WORKSHEET NAME | CELL IDENTIFIER | CELL FORMULA |
| 8 |  |  |  |
| 9 | Cob Input | B6 | =3/8 |
| 10 | Cob Input | E9 | ='Truck Size vs Plant Size'!G25*2 |
| 11 | Cob Input | H9 | =(600000*B8/1000)+((E12-600000)*H2Ocost/1000) |
| 12 | Cob Input | B10 | =31520/(40*52) |
| 13 | Cob Input | E12 | =ABS(((feedrate*MCWBf)/(1-MCWBf))-((feedrate*MCWBi)/(1-MCWBi))*B29/B28)*B16/24 |
| 14 | Cob Input | H12 | =Production_Labor_Requirement*employeepayrate*40*52 |
| 15 | Cob Input | B15 | =B16*(2/3) |
| 16 | Cob Input | B16 | =24*B14/2 |
| 17 | Cob Input | H17 | =H16/(feedrate*B16/24) |
| 18 | Cob Input | E32 | =tot_mod_cost*0.3 |
| 19 | Equipment Cost and Size | C4 | =((feedrate*MCWBi/(1-MCWBi))+feedrate)/24 |
| 20 | Equipment Cost and Size | C9 | ='Truck Size vs Plant Size'!G25*2 |
| 21 | Equipment Cost and Size | E29 | =(((E27/(10*2.54*12))*E28)/9)*E26 |
| 22 | Equipment Cost and Size | B30 | =10*2.54*12*3 |
| 23 | Equipment Cost and Size | E35 | =E33*(((E34/(12))*E28)/9) |
| 24 | Equipment Cost and Size | B36 | =(((feedrate*MCWBf)/(1-MCWBf))-(feedrate*MCWBi/(1-MCWBi)))*'Cob Input'!B30/(24*60) |
| 25 | Truck Size vs Plant Size | B6 | =19*(96/12)*5 |
| 26 | Truck Size vs Plant Size | B12 | =60/(B9+B7) |
| 27 | Truck Size vs Plant Size | B13 | =60/(B10+B9) |
| 28 | Truck Size vs Plant Size | B14 | =ROUNDUP((B7/(B9+B10))+1, 1) |
| 29 | Truck Size vs Plant Size | B16 | =(B$4*A16/(1+B$4))+A16 |
| 30 | Truck Size vs Plant Size | C16 | =B16/24 |
| 31 | Truck Size vs Plant Size | F16 | =ROUNDUP((E16/B$13),0) |
| 32 | Truck Size vs Plant Size | G16 | =(ROUNDUP(F16,0))*ROUNDUP(B$14,0) |

**Figure 2. Screenshot of the Hard-Coding Error Report (HCER). Summary error statistics are shown at top of page, while specific error instances are listed below. Unity cells (unique) are marked with a grey fill.**


Latest_HCD&SubCat_2011 - Microsoft Excel

E5 — Frequency of Other Unidentified Numerals =

| | D | E | F | G | H |
|---|---|---|---|---|---|
| 1 | | SUBCATEGORISATION OF HCE : | | | |
| 2 | | Frequency of Unity Errors = 14% | | | |
| 3 | | Frequency of Power_of_10 Conversions = 14% | | | |
| 4 | | Frequency of Unit Conversion Factors= 18% | | | |
| 5 | | Frequency of Other Unidentified Numerals = 54% | | | |
| 6 | | | | | |
| 7 | | # OF UNITY OCCURENCES | # OF POWER_OF_10 CONVERSIONS | # OF UNIT CONVERSION FACTORS | # OF OTHER UNIDENTIFIED NUMERALS |
| 8 | | | | | |
| 9 | | 0 | 0 | 0 | 2 |
| 10 | | 0 | 0 | 0 | 1 |
| 11 | | 0 | 2 | 0 | 2 |
| 12 | | 0 | 0 | 0 | 3 |
| 13 | | 2 | 0 | 1 | 0 |
| 14 | | 0 | 0 | 0 | 2 |
| 15 | | 0 | 0 | 0 | 2 |
| 16 | | 0 | 0 | 1 | 1 |
| 17 | | 0 | 0 | 1 | 0 |
| 18 | | 0 | 0 | 0 | 1 |
| 19 | | 1 | 0 | 1 | 0 |
| 20 | | 0 | 0 | 0 | 1 |
| 21 | | 0 | 1 | 1 | 2 |
| 22 | | 0 | 1 | 1 | 2 |
| 23 | | 0 | 0 | 0 | 2 |
| 24 | | 2 | 0 | 2 | 0 |
| 25 | | 0 | 0 | 0 | 4 |
| 26 | | 0 | 0 | 1 | 0 |
| 27 | | 0 | 0 | 1 | 0 |
| 28 | | 2 | 0 | 0 | 0 |
| 29 | | 1 | 0 | 0 | 0 |
| 30 | | 0 | 0 | 1 | 0 |
| 31 | | 0 | 0 | 0 | 1 |
| 32 | | 0 | 0 | 0 | 2 |

Energy Usage Summary / Belt Conveyor Design / Screw Conveyor Design / Labor Calculations / HCER

Ready

11:16 PM Wednesday 5/11/2011

**Figure 3. Screenshot of the results of sub-categorization of hard-coding errors. Summary statistics are shown at top of page, while a matrix of instances of each subcategory is shown below.**

## Audit Results from Bioenergy-relevant Spreadsheets

The results of the audits are shown in tables 2 and 3. Table 2 shows the Cell Error Rate (CER) of hard-coding errors in the tested spreadsheets ranged from 11% to 44%. The workbook with the lowest CER (FEBEF, 11%) originally had a 45% CER; the 11% reported reflected a major effort to remove hundreds of instances of hard-coding errors. If we had not actively improved FEBEF based on the audit, the minimum observed CER would have been 22% (GBAMM and Ethanol Profitability). While eliminating HCE instances from FEBEF, one of the co-authors of this article found a hard-coded cell which also contained a serious mathematical representation error, that caused significant mistakes in the bottom-line values in that spreadsheet. This reveals yet another facet of hard-coding errors – namely their ability to mask other kinds of errors and consequently, be damaging to the spreadsheet. Although we only explored six spreadsheets, a total of nearly 72,000 formulae cells were checked. Interestingly, but perhaps not surprisingly, we observed HCE frequencies similar to the 43.5% reported by Powell et al. (2009).

**Table 2. Cell error rates (CER) of hard-coding errors (HCE) in the six tested spreadsheets.**

| Workbook Tested | Total No. of Cells Checked | No. of Cells with HCE | CER (%) |
|---|---|---|---|
| Cob Cost | 203 | 89 | 44 |
| GBAMM | 462 | 100 | 22 |
| S-FAAD | 702 | 181 | 26 |
| FEBEF | 844 | 90 | 11 |
| Ethanol Profitability | 2757 | 608 | 22 |
| GREET | 66945 | 26867 | 40 |

Table 3 provides distribution statistics on the hard-coding errors in the six spreadsheets. The values in table 3 are the frequencies of each type as a percentage of the total hard-coding instances in the respective workbooks. Figure 4 provides a pie-chart representation of frequencies of each subcategory of hard-coding error for all six spreadsheets put together. For all the six spreadsheets, subtotals of instances of unity errors, power of 10 conversions, unit conversions, and other unidentified numerals were computed. The total count of hard-coding errors was obtained by summing up the four subtotals. Next, the frequencies of each of the subcategories were calculated as a percentage of the total instances of hard-coding errors.

**Table 3. Sub-categorization of hard-coding errors (HCE) from six tested spreadsheets.** [a]

| Workbook Tested | Unity Errors (%) | Power of 10 (%) | Commonly Used Unit Conversions (%) | Other Unidentified Numerals (%) |
|---|---|---|---|---|
| Cob Cost | 14 | 14 | 18 | 54 |
| GBAMM | 7 | 69 | 4 | 20 |
| S-FAAD | 21 | 7 | 46 | 26 |
| FEBEF | 94 | 0 | 0 | 6 |
| Ethanol Profitability | 8 | 77 | 15 | 0 |
| GREET | 47 | 25 | 9 | 19 |

[a] Each of the percentages is specific to the spreadsheet, i.e., 14% unity errors mean 14% of the total number of HCE instances in Cob Cost were unity errors

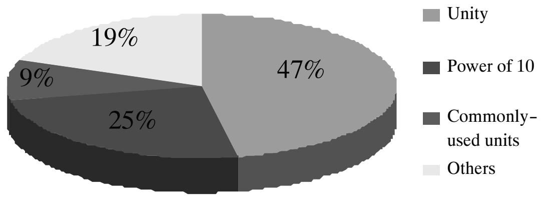


**Figure 4. Distribution of sub-categories of HCE, showing preponderance of unity and power of 10 errors in the spreadsheets tested.**

Both GBAMM and Ethanol-Profitability workbooks suffered from high rates of Power of 10 conversions (69% and 77%, respectively, of the total instances of HCE), which justifies their separate categorization from unit conversions. Reflecting the effort to rid FEBEF of power of ten and unit conversion errors, unity errors predominate in FEBEF (at a rate of 94% of the total instances of HCE in FEBEF). The Cob Cost, S-FAAD and GREET had unity errors exceeding 14%, while the Ethanol-Profitability Workbook and GBAMM had fewer than 10%. Other unidentified numeral hard-coding errors formed a significant mass of errors in most of the spreadsheets with a frequency reaching as high as 54% of total HCE instances in the Cob-Cost Workbook. In some instances, their frequency even exceeds those of the commonly used unit conversion factors. Future versions of this program could allow users to specify additional numerical values used heavily in their spreadsheets.

To reduce the frequency of hard-coding errors and their impacts, spreadsheet authors can create an "Assumptions" tab in the beginning of the spreadsheet. By listing necessary conversions and other important constants used during the course of development of the spreadsheet, and then assigning them a brief but descriptive moniker (e.g., "Acresperha," "rhoH2O" etc.) using the "define name," functionality in Excel, one can get rid of hard-coding errors substantially. When dealing with unit conversions in particular, users can also use the built-in "CONVERT" function of Microsoft Excel. Using 3.875 L/gal in one cell and 3.785 L/gal (the correct value) in another cell of the same spreadsheet leads to a 2.4% quantitative error, and constitutes a duplication error because the same value is being coded as multiple values in the spreadsheet. The "CONVERT" function will help to maintain consistency throughout the workbook and duplication errors can be avoided.

## Conclusions

The frequency of hard-coded cells or the CER of hard-coding errors in the tested bioenergy-relevant spreadsheets ranged from 11% to 44%. This turns out to be a high error rate, especially since each occurrence is an opportunity for more serious numerical errors. We recommend the replacement of hard-coded values by unique descriptive monikers, as discussed before. By systematically using these named factors in equations, most hard-coding errors can be eliminated. Factors that occur rarely, perhaps in only one or two cells, can similarly be replaced by a named factor, but the cost-benefit ratio is questionable. Having a small fraction (e.g., less than 1%) of cells with such errors is probably not a major problem for most spreadsheets, especially if an auditing program such as the ones describe here are used to rapidly review any HCE instances. Along with structuring spreadsheets to make computations easy to follow, and clearly listing units on all quantities, elimination (or at least minimization) of hard-coding errors must be considered another fundamental part of good spreadsheet practices.

### Acknowledgements

Partial financial support for this work was provided by the USDA Higher Education Challenge Grant Award #2006-38411-17034. The authors would like to thank Carol Faulhaber and Sami Khanal for the technical guidance provided through the course of this project.

## References

EUSPRIG. (n.d.). European Spreadsheet Risks Interest Group stories. Available at: www.eusprig.org/stories.htm.

Galletta, D. F., K. S. Hartzel, S. E. Johnson, J. L. Joseph, and S. Rustagi. 1997. Spreadsheet presentation and error detection: An experimental study. *J. Mgmt. Info. Systems* 13(3): 45-63.

Panko, R. R., and R. P. Halverson. 1996. Spreadsheets on trial: A survey of research on spreadsheet risks in *Proceedings of the 29th Hawaii International Conference on System Sciences (HICSS),* Wailea, Hawaii 2: 326-335.

Panko, R. R. 1999. Applying code inspection to spreadsheet testing. *J. Mgmt. Info. Systems* 16(2): 159-176.

Powell, S. G., K. R. Baker, and B. Lawson. 2008. A critical review of the literature on spreadsheet errors. *Decision Support Systems* 46(1): 128-138.

Powell, S. G., K. R. Baker, and B. Lawson. 2009. Errors in operational spreadsheets. *J. Organizational and End User Computing* 21(3): 24-36.

Rajalingham, K., D. R. Chadwick, and B. Knight. 2008. Classification of spreadsheet errors. *Computing Res. Repository* abs/0805.4224.

Teo, T. S. H., and J. E. Lee-Partridge. 2001. Effects of error factors and prior incremental practice on spreadsheet error detection: An experimental study. *The Intl. J. Mgmt. Sci.* 29(5): 445-456.