"Good" Can Mean So Many Things

Dianne P. O'Leary

This case study investigates various meanings of the word good when applied to an algorithm. We consider speed and correctness of a program called `slow.m`. This case study is closely related to Chapter 4 in the SCCS textbook.

First we study the program as given. Then we try to improve the program.

CHALLENGE 0.1. Consider the legacy software `slow.m` in Table 1 and on the website.

(a) Add clear documentation to `slow.m`, describing the program's purpose, input, output, and the method, listing the author as unknown, and listing you as documenter, with date.

(b) Judge `slow.m` for its clarity (good variable names, well-formatted listing), modularity, correctness, reliability, and efficiency.

(c) What does the program do?

(d) Develop a testing program for `slow.m`

CHALLENGE 0.2.

(a) Modify `slow.m` to make it run as fast as possible.

(b) Were you completely successful in making the algorithm column oriented? If not, what data accesses fail to be column oriented?

(c) Using real arithmetic, under what condition does the algorithm fail to compute finite values for the entries in the matrix $\boldsymbol{X}$? Answer the same question for floating point arithmetic.

Table 1. *Function* `slow.m`

```
function x = slow(a,b,c)

[m,n] = size(a);

x(1,1) = c(1,1)/(a(1,1)+b(1,1));

for i=1:n,
  x(1,i) = c(1,i);
  for j=1:i-1,
    x(1,i) = x(1,i) - x(1,j)*a(j,i);
  end
  temp = b(1,1)+a(i,i);
  x(1,i) = x(1,i) / temp;
end

for k=2:n,
   for i=1:n,
     x(k,i) = c(k,i);
     for j=1:i-1,
       x(k,i) = x(k,i) - x(k,j)*a(j,i);
     end
     for j=1:k-1,
       x(k,i) = x(k,i) - x(j,i)*b(k,j);
     end
     temp = b(k,k)+a(i,i);
     x(k,i) = x(k,i) / temp;
   end
end
```

POINTER 0.1. Approaching undocumented code
The lack of documentation in `slow.m` makes it very hard to figure out what it does. Perhaps the best approach is to "play computer": try writing out the results for $n = 3$. The fundamental question is, what equation does the matrix $\boldsymbol{X}$ satisfy? To answer this, try to put the $n = 3$ results together so that you get an equation for $\boldsymbol{X}$ in terms of $\boldsymbol{A}$, $\boldsymbol{B}$, and $\boldsymbol{C}$, without using any matrix inverses.

In completing this case study, you illustrate several definitions of good software:

- **correct**.
- **efficient in time used**.

POINTER 0.2. Optimizing computer code
In the early days of computing, a good algorithm had to take a small amount of time (since computers had a small mean-time-to-failure) and use a small amount of memory (since both program and data had to fit into an incredibly small physical memory). People went to great lengths to reduce the number of divisions in an algorithm, because on early machines, that operation was much slower than addition, subtraction, or multiplication.
Later, people tried to minimize the number of floating point computations, since this was a good measure of time.
More recently, the time for a floating point computation is dominated by the time to access the data in memory, so the number of data accesses has become the focus of algorithm optimization.
Most recently, people have become concerned about power consumption. We may soon be back to reducing the number of divisions in our algorithms, since power consumption is greater for division than for addition, subtraction, or multiplication!

- **well-documented**.
- **easy-to-use**.
- **as simple as possible**.

There are several definitions other than these, for example,

- **robust** in terms of checking for errors in the input. For example, we should check that the dimensions of the input matrices are consistent.
- **efficient in storage** used.
- **low in power consumption**. If you hold your laptop on your lap, this is related to how much heat you feel as the program runs. Although keeping your lap cool may not be a high priority in algorithm design, power consumption is becoming a very important design parameter, since it affects how many problems can be solved between battery charges for a remote or wearable device, or the size of a solar panel needed to run a device, or the amount of air conditioning needed for a room full of computers.