Improving performance

Matlab’s a pig

Outline

• Announcements:
  – Homework I: Solutions on web
  – Homework II: on web--due Wed.
• Homework I
• Performance Issues

Homework I

• Grades & comments are waiting in your mailboxes
  – PASS--you passed!
  • try to learn from your mistakes
  – PROVISIONAL--you passed, but I’m watching
  • 2 or more provisional passes will make it difficult for me to let you pass
  – FAIL--you’re failing and need to see me ASAP
  • No one yet!
Homework I

- Most everyone did well on 1-4
  - check the comments I sent
- 5 and 6 gave people fits
  - Need to understand the solutions in order to solve next assignment.

Homework

- "Essential knowledge" questions should be fairly easy
  - just the basics covered in lecture
- "Programming" questions will be harder
  - apply what we've talked about to a real problem
- The goal of the problem sets is to build your skill and confidence
  - I don't intend for this to be painful
  - If you find that you're spending several hours on a problem, please see me.

Fourier Series--Problems 5-6

- We can represent a function $x(t)$ by sines & cosines
- Define a vector of times $t$ for which we want to know the value of $x$.
  - $t$ and $x$ are vectors of the same size.
- The $j$th entry in $x$ will be its value at time $t(j)$ is given by:
  $$x(j) = \sum_{k=-N/2}^{N/2-1} a_k \cos \left( \frac{2\pi (k-1) t(j)}{N \Delta t} \right) + b_k \sin \left( \frac{2\pi (k-1) t(j)}{N \Delta t} \right)$$
Fourier Series--Problems 5-6

• To keep things simple, let’s ignore sine terms and pretend the cosines don’t exist:

\[ x(t) = \sum_{n=0}^{N} a_n \cos\left(\frac{2\pi(n-1)t}{N}\right) \]

• Can implement this as a double loop:

```matlab
n = length(a); N = 2*n-2; p = length(t);
x = zeros(p,1); f = 2*pi/(N*dt);
for j = 1:p
    for k = 1:n
        x(j) = x(j) + a(k)*f*(k-1)*t(j);
    end
end
```

Fourier Series--Problems 5-6

• Inner loop looks a lot like a vector product c = a'*b:

```matlab
c = 0;
for k = 1:n
    c = c + a(k)*b(k);
end
```

• Can eliminate inner loop:

```matlab
n = length(a); N = 2*n-2; p = length(t);
x = zeros(p,1); f = 2*pi/(N*dt);
K = (1:n)-1;
for j = 1:p
    x(j) = f*t(j)*K*a(:);
end
```

Fourier Series--Problems 5-6

• If we can use vector * to eliminate one loop, why not the other?
  - Multiplying t & K gives a p-by-n matrix in which each row is K scaled by an element of t:
    ```matlab
t(1)*K = [t(1)*K; t(2)*K; ...; t(p)*K]
```
  - If we multiply this matrix by a, we get the desired form for x:
    ```matlab
t(1)*K*a(j) = [t(1)*K*a; t(2)*K*a; ...; t(p)*K*a]
```
**Problem Set II**

- You must implement the scheme we developed as a function
  - Inputs: a, b, t
  - Outputs: x

- You will create another function that will solve for a and b
  - Inputs: x, t
  - Outputs: a, b, f

**So what’s the point?**

- Matrix operators in Matlab are much faster than loops
- Example developed above:
  - TwoLoop.m
  - OneLoop.m
  - NoLoop.m
- Fast Matlab code uses * and avoids loops

**Some Performance Tips**

- Use built-in functions as they are often heavily optimized
  - Vectorization is the epitome of this
- Minimize division
  - \(x/2\) takes longer than 0.5*x
- Do computations outside loop
  - f and K in TwoLoop.m
- Pre-allocate arrays
  - for \(j=1:n; a(j)=<\text{something}>;\) end
  - Setting \(a=\text{zeros}(1,n)\) before the loop speeds things up
Other Options

- subfunctions
  - file fname.m:
    - function O=fname(I)
    - function O2=fname2(I2)
- Implement in a compiled language
  - mapping to C and Fortran is straightforward
- Look into Matlab compiler
- Stop being so impatient

Some comments on performance

- The Three “E’s”
  - Effective
  - Efficient
  - Elegant
- Efficiency (speed) is only one goal.
- Time spent tuning code should be factored into performance
  - Spending 2 hours improving runtime from 10 min to 5 min only makes sense if you will use the code a lot or on much larger problems