ODE IVPs/BVPs
AND A NEW TEXTBOOK

Nick Trefethen, Oxford

chebfun

Purpose: Chebfun is a tool for numerical computing with functions.

Idea: overload Matlab's vectors/matrices to functions/operators.

Algorithms: based on piecewise Chebyshev polynomial interpolants.

www.chebfun.org

BVP

u = L\f

This is Chebfun's syntax for solving ODE BVPs $Lu = f$.
L can be linear or nonlinear and includes BCs.

Inputs: f is a chebfun, L is a chebop.
Output: u is a chebfun.

Algorithm: adaptive Chebyshev spectral collocation embedded in a Newton iteration. Formulation via *block operators* $\longrightarrow$ rectangular matrices (Driscoll + Hale)

IVP

What about IVPs ? Here Chebfun's algorithm is completely different: marching with **`ode113`**, then conversion of the result to a chebfun.

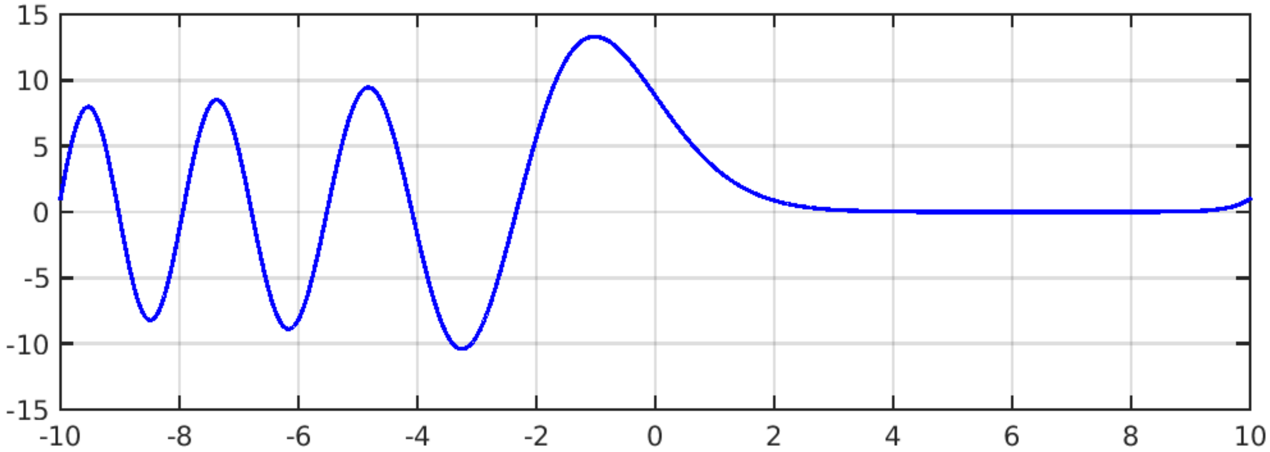
Until 2015, you had to call **`chebfun.ode113`**.
In 2014, Birkisson folded this into the syntax u = L\f.
This was nontrivial because higher-order problems must be converted silently to first order for **`ode113`**.

So now, Chebfun solves all ODE problems with u = L\f .

Airy equation (scalar linear BVP)

$y'' - xy = 0, \quad x \in [-10,10], \quad y(-10) = y(10) = 1.$

```
L = chebop(-10,10);
L.op = @(x,y) diff(y,2) - x*y;
L.lbc = 1; L.rbc = 1;
y = L\0; plot(y)
```



[0.2 secs.]

```
y(0)
ans = 8.82249323
```

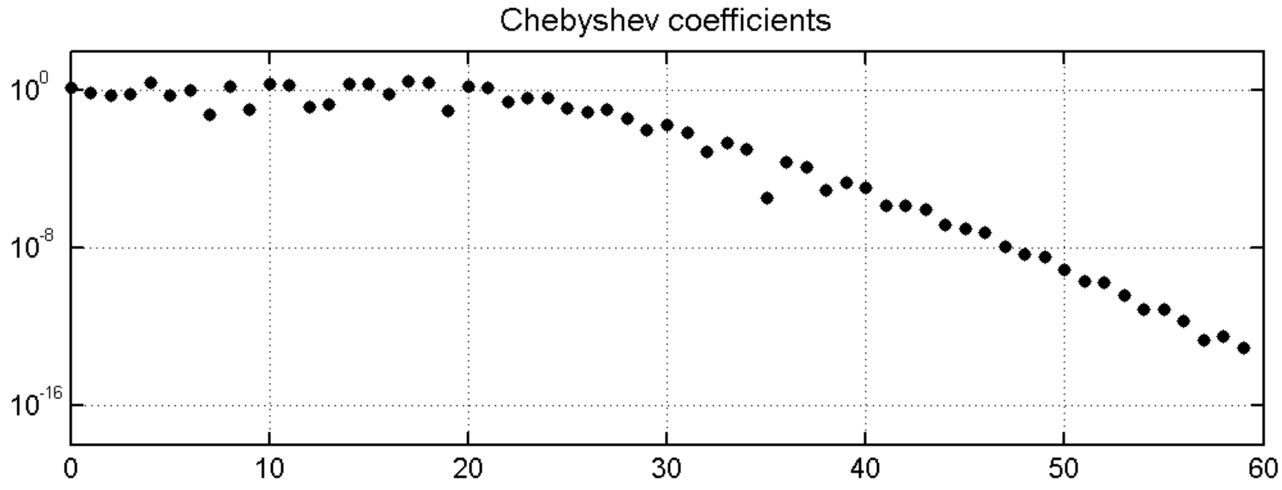
```
max(y)
ans = 13.31113744
```

Airy equation (scalar linear BVP)

$y'' - xy = 0, \quad x \in [-10,10], \; y(-10) = y(10) = 1.$

```
L = chebop(-10,10);
L.op = @(x,y) diff(y,2) - x*y;
L.lbc = 1; L.rbc = 1;
y = L\0; plot(y)
```

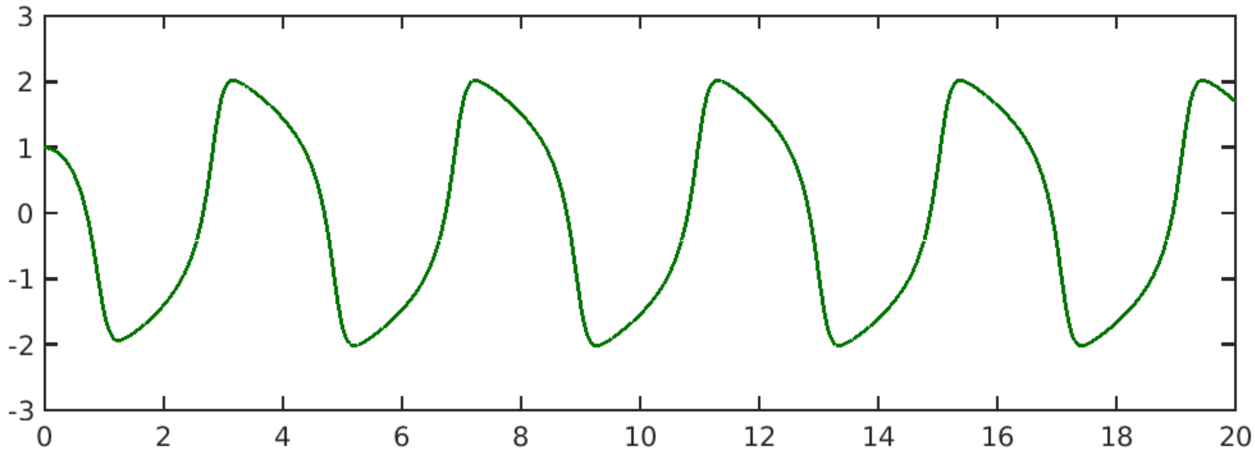
```
plotcoeffs(y,'.'), ylim([1e-18,1e2])
```



van der Pol equation (scalar nonlinear IVP)

$0.3y'' - (1 - y^2)\,y' + y = 0, \quad t \in [0,20], \;\; y(0) = 1, \;\; y'(0) = 0.$

```
N = chebop(0,20);
N.op = @(t,y) 0.3*diff(y,2) - (1-y^2)*diff(y) + y;
N.lbc = @(y) [1; 0];
y = N\0; plot(y)
```



[0.6 secs.]

```
[val,pos] = max(y,'local');
diff(pos)
```

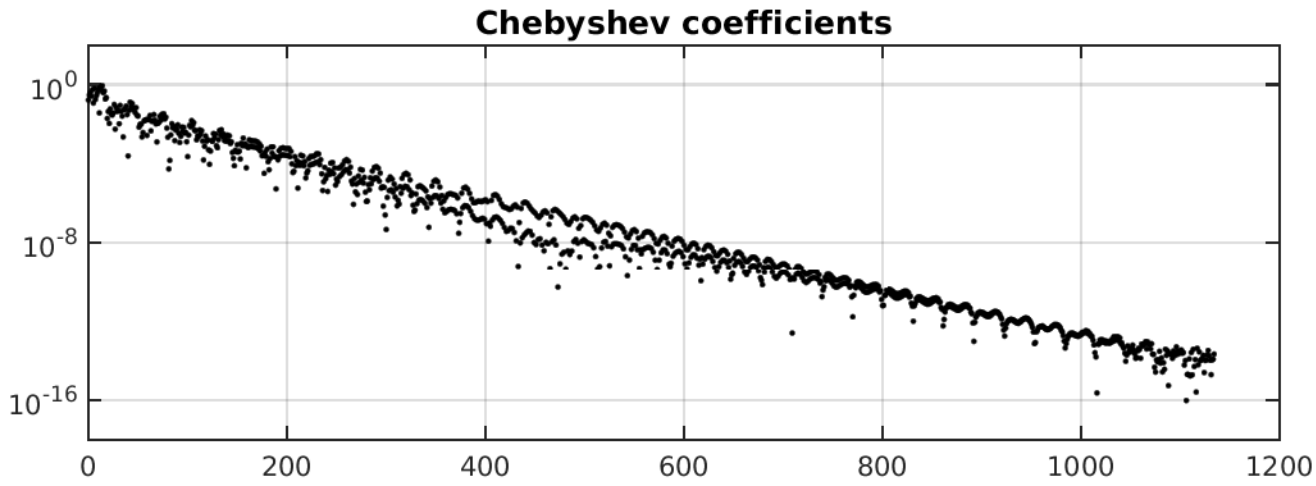
```
ans = 3.16508501
      4.07245435
      4.07253777
      4.07253777
      4.07253777
```

van der Pol equation (scalar nonlinear IVP)

$0.3y'' - (1-y^2)y' + y = 0, \quad t \in [0,20], \; y(0) = 1, \; y'(0) = 0.$

```
N = chebop(0,20);
N.op = @(t,y) 0.3*diff(y,2) - (1-y^2)*diff(y) + y;
N.lbc = @(y) [1; 0];
y = N\0; plot(y)
```

```
plotcoeffs(y,'.'), ylim([1e-18,1e2])
```

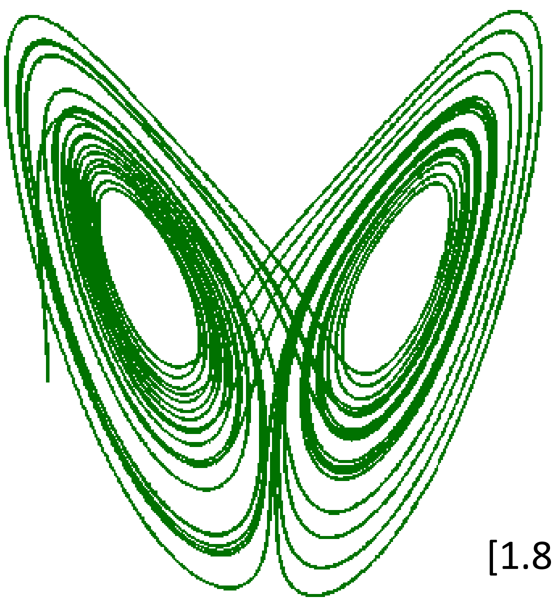


Lorenz equations (system of nonlinear IVPs)

$$u' = 10(v-u),\ \ v' = u(28-w)-v,\ \ w' = uv-(8/3)w,$$

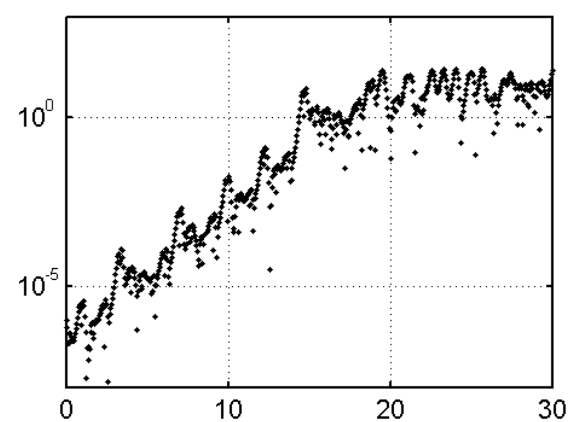
$$t \in [0,30],\ \ u(0) = v(0) = -15,\ w(0) = 20.$$

```
N = chebop(0,30);
N.op = @(t,u,v,w) [diff(u)-10*(v-u); ...
           diff(v)-u*(28-w)+v; diff(w)-u*v+(8/3)*w];
N.lbc = @(u,v,w) [u+15; v+15; w-20];
[u,v,w] = N\0; plot(u,w)
```



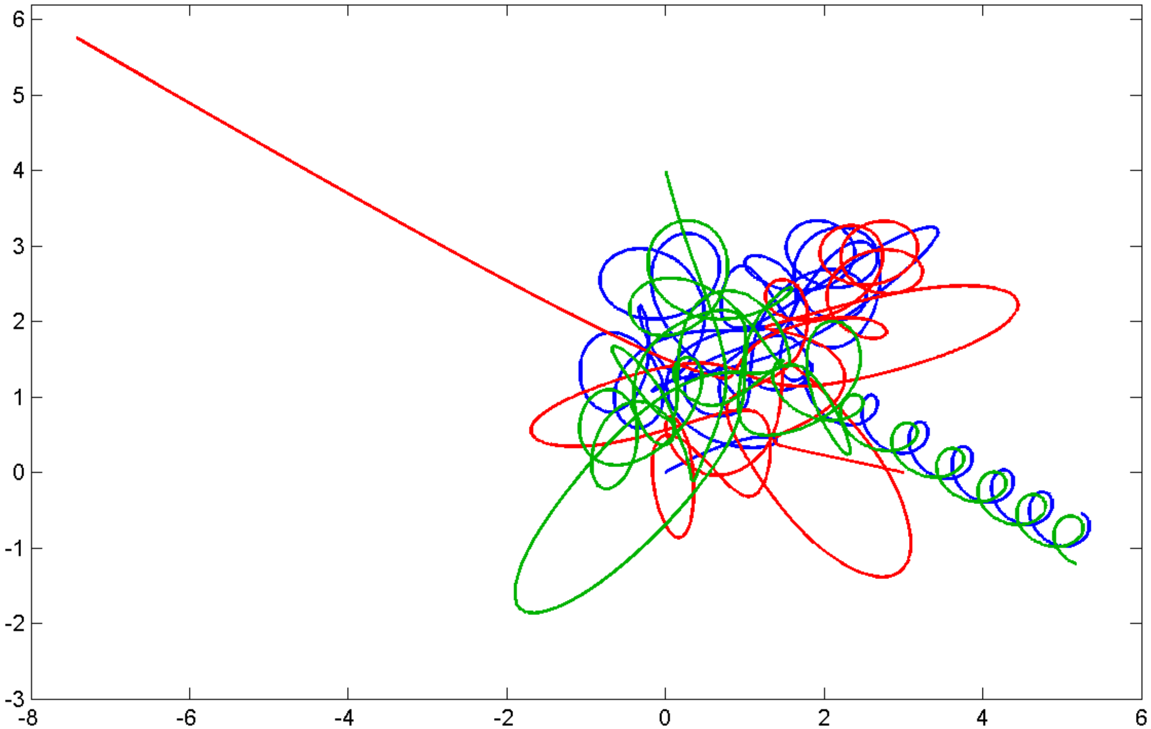
[1.8 secs.]

```
N.lbc = @(u,v,w) [u+15.000001;v+15;w-20];
[u2,v2,w2] = N\0;
tt = 0:.05:30;
semilogy(tt,abs(u2(tt)-u(tt)),'.')
```



3-body problem (system of nonlinear IVPs)

```
N = chebop(0,100); u0 = 0; v0 = 3; w0 = 4i;
N.op = @(t,u,v,w) [ ...
    diff(u,2) + (u-v)/abs(u-v)^3 + (u-w)/abs(u-w)^3; ...
    diff(v,2) + (v-u)/abs(v-u)^3 + (v-w)/abs(v-w)^3; ...
    diff(w,2) + (w-u)/abs(w-u)^3 + (w-v)/abs(w-v)^3];
N.lbc = @(u,v,w) [u-u0; v-v0; w-w0; diff(u); diff(v); diff(w)];
y = N\0; plot(y)
```



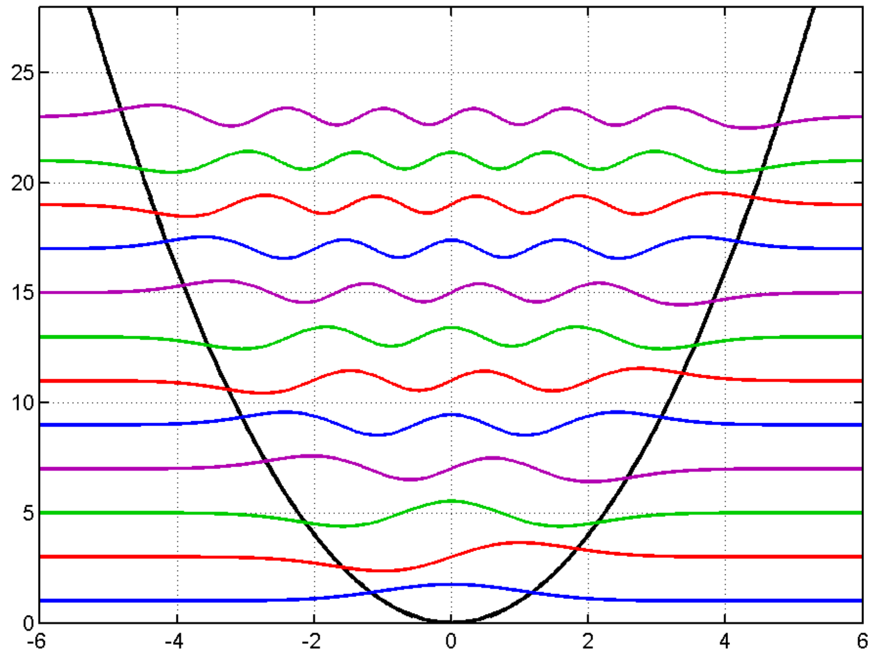

```
roots(real(v)+2)
ans = 90.1543
```

[9.0 secs.]

Harmonic oscillator (scalar linear eigenvalue problem)

$$-y'' + x^2 y = \lambda y, \quad x \in [-6,6], \; y(-6) = y(6) = 0.$$

```
L = chebop(-6,6);
V = chebfun('x.^2',[-6,6]);
plot(V,'k'), hold on
L.op = @(x,y) -diff(y,2) + V*y;
L.lbc = 0; L.rbc = 0;
[Y,D] = eigs(L,12);
for k = 1:12, plot(Y(:,k)+D(k,k)), end
```



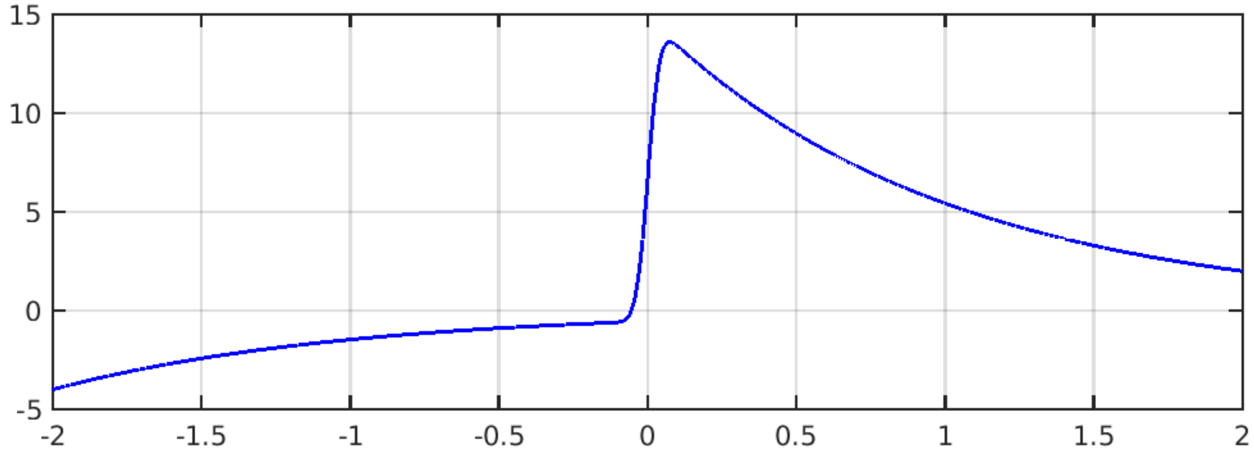

```
d = diag(D); d(1:5)
ans = 1.0000000
      3.0000000
      5.0000000
      7.0000000
      9.0000000
```

[1.0 secs.]

Interior layer equation (scalar linear BVP)

$0.001y'' + xy' + xy = 0, \quad x \in [-2,2], \;\; y(-2) = -4, \;\; y(2) = 2.$

```
L = chebop(-2,2);
L.op = @(x,y) 0.001*diff(y,2) + x*diff(y) + x*y;
L.lbc = -4; L.rbc = 2;
y = L\0; plot(y)
```



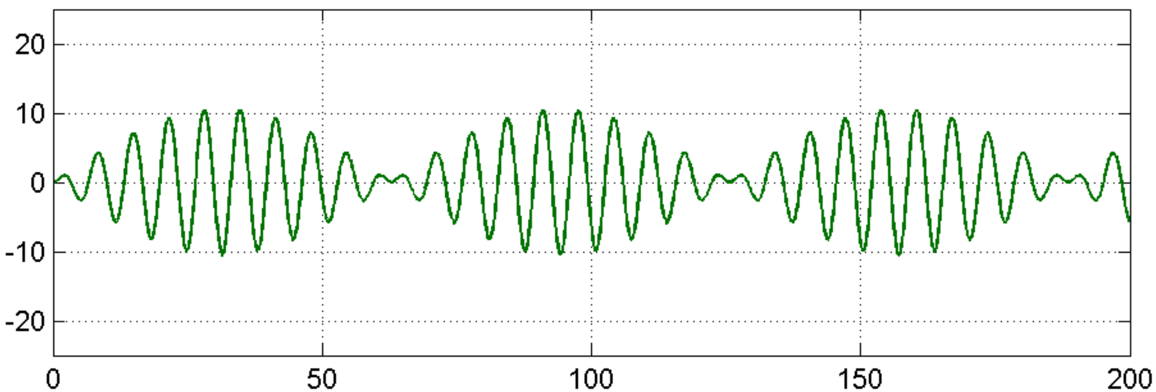
[0.9 secs.]

```
deriv(y,0)
ans = 186.918317
```

Resonance (scalar linear IVP)

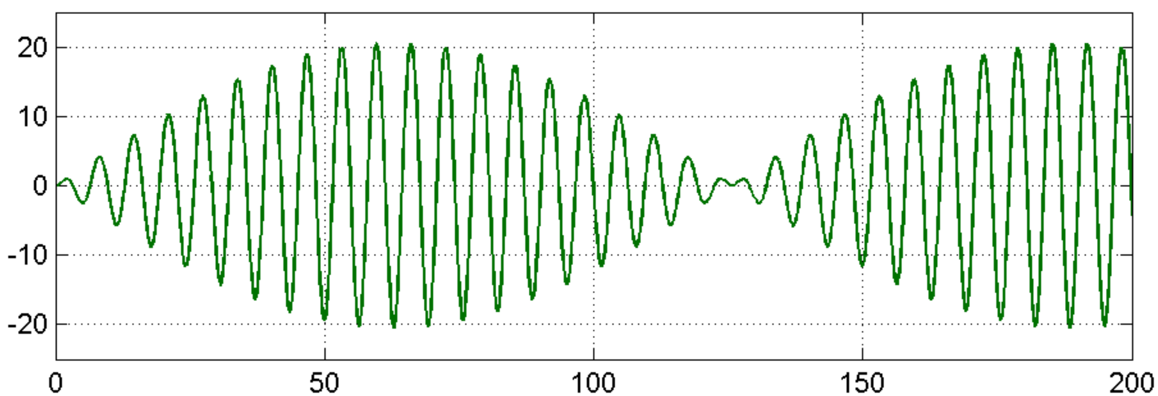
$y'' + y = 0, \quad t \in [0,200], \; y(0) = y'(0) = 0.$

```
t = chebfun('t',[0 200]);
L = chebop(0,200);
L.op = @(t,y) diff(y,2) + y;
L.lbc = [0; 0];
```




```
y = L\cos(.9*t)
plot(y)
norm(y)
ans =  72.7106
```

[2.9 secs.]

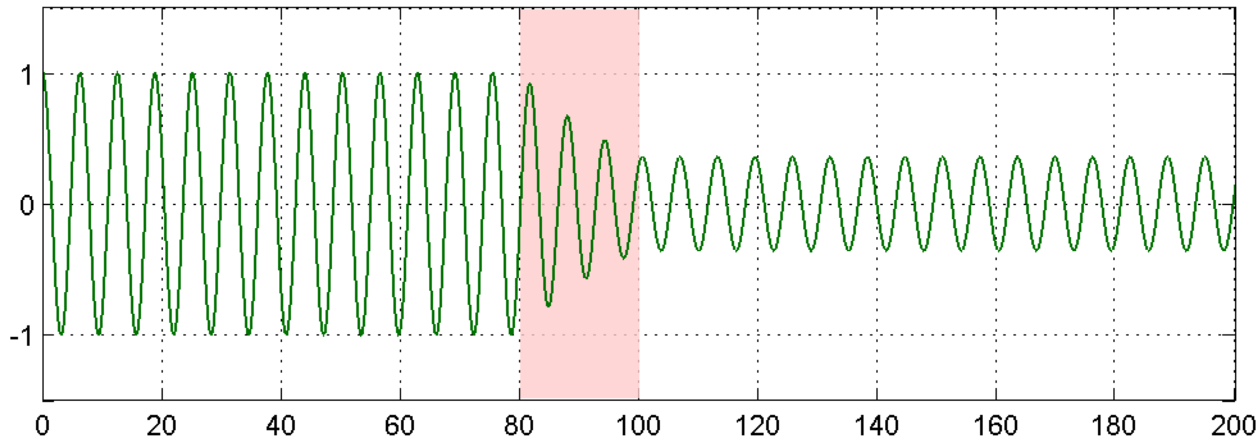



```
y = L\cos(.95*t)
plot(y)
norm(y)
ans = 148.8003
```

Damping switched on/off (scalar linear IVP, discontinuous)

$y'' + d(t)y' + y = 0, \quad t \in [0,200], \; y(0)=1, \; y'(0) = 0.$

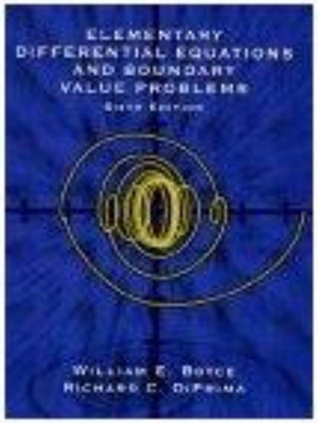
```
t = chebfun('t',[0 200]);
d = 0.1*(abs(t-90)<10);
L = chebop(0,200);
L.op = @(t,y) diff(y,2) + d*diff(y) + y;
L.lbc = @(y) [1; 0];
y = L\0; plot(y)
```

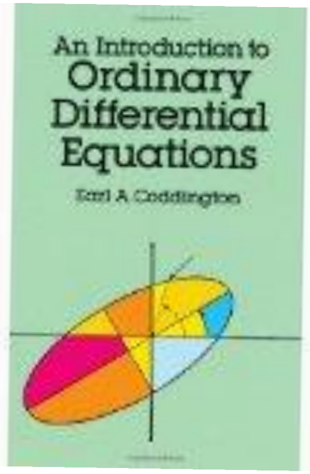


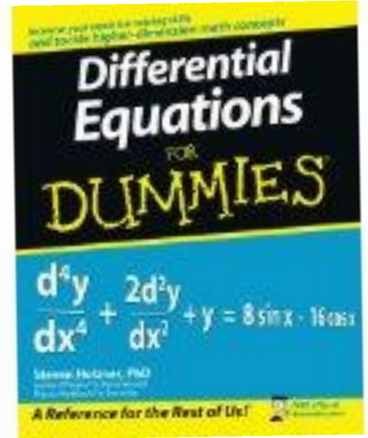

```
max(y{0,50})
ans = 1.0000000
max(y{150,200})
ans = 0.3582228
```

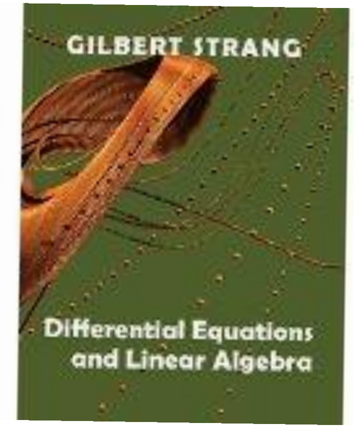
[3.2 secs.]

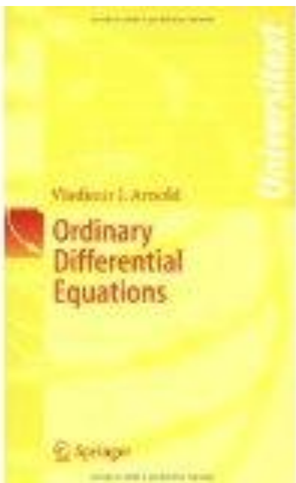
Even easier: Birkisson’s graphical user interface chebgui.

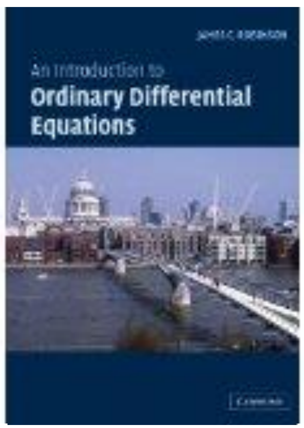




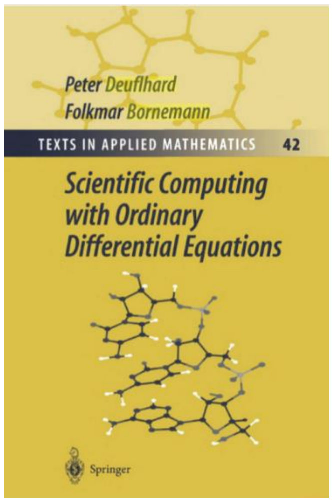


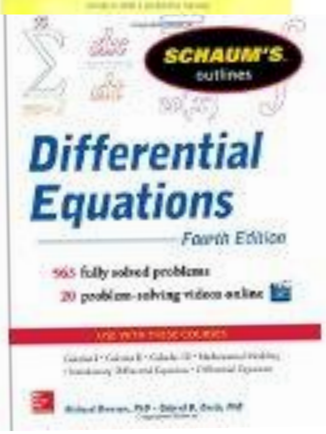


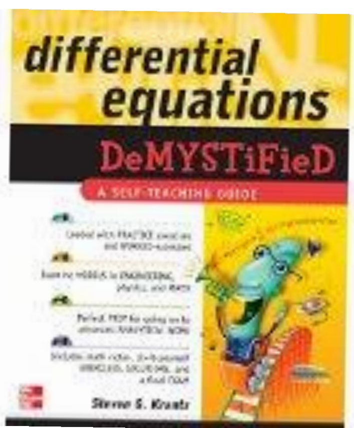


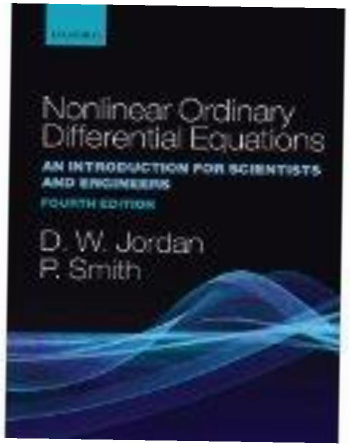


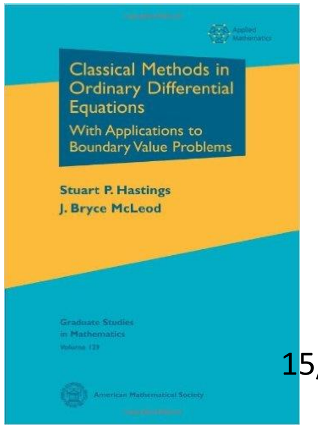
There are lots of ODE books.











- Our aim is to write a book quite unlike the others:
  focusing on ODE *behavior*, illustrating everything effortlessly.
- Enabled by numerics, but not a numerical book.
- Mathematically mature, but not technically advanced.
- Nonlinear problems and BVPs throughout.
- A book that certain instructors will choose to teach a course from, and they will all want to look at.   A book that the top 10% of students in any ODE course will be excited to discover.
- Cheap from SIAM, and freely available online.

Exploring ODEs

Lloyd N. Trefethen, Ásgeir Birkisson, and Tobin A. Driscoll

Still to be written:
· Linearization and classification of fixed points
· Bifurcation
· Appendix B: 100 more examples

Draft LNT 15 January 2016. Final book will be 19+2 chapters, ≈ 260 pages.

Please email me pointers to good examples and any other suggestions.