Numerical Integration of ordinary D.E.'s

I'll continue to use $\underline{x}$ as dependent variable and t as independent variable.

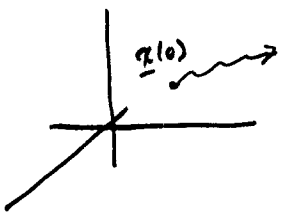
STATE SPACE FORM: $\dot{\underline{x}} = \underline{f}(\underline{x}, t)$ $\underline{x}$ n-vector, fctn. of t

Or write

$$\begin{cases} \frac{dx_1}{dt} = f_1(x_1, \cdots, x_n, t) \\ \frac{dx_2}{dt} = f_2(x_1, \cdots, x_n, t) \\ \vdots \\ \frac{dx_n}{dt} = f_n(x_1, \cdots, x_n, t) \end{cases}$$

Boundary Conditions Can be very complicated
two simple cases cover most practical situations

initial-values given $x_1(0), x_2(0), \cdots, x_n(0)$



two-point values given Usually some at $t=0$, some at $t = T_{final}$.
(harder)

---

will stick to initial-value problems.

Often these arize from situation like

$$\frac{d^2y}{dt^2} + \alpha(t)\frac{dy}{dt} + \beta(t)y = 0$$

& initial conditions $y(0)$ and $y'(0)$
(equivalent to $x_1(0)$ and $x_2(0)$ when converted to state-space)

We'll look at following general methods

1) Euler's Method (simple-minded, basis of many other methods, not good by itself)

2) predictor-corrector methods (can be excellent for reasonable problems)

3) Runge-Kutta (usually implies 4th order) (good workhorse - will be good enough for assignment 3, but not state-of-the-art)

See Num. Recipes & refs therein for more sophisticated methods, like Richardson extrapolation. Integrating ODE's numerically is a grown-up problem!

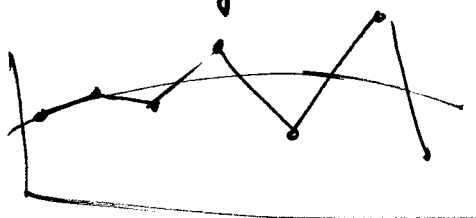
---

three criteria enter into evaluating a numerical integration method: (besides complexity of coding, of course)

1) accuracy
- estimated using standard Taylor series arguments, $O(h^2)$, $O(h^3)$ ... etc.
- plus empirical evidence

2) efficiency
- running time can be difficult to predict, since we usually don't know $h = \Delta t$ in advance, and it may be chosen adaptively in practice during run-time

3) stability
- some methods diverge on some problems



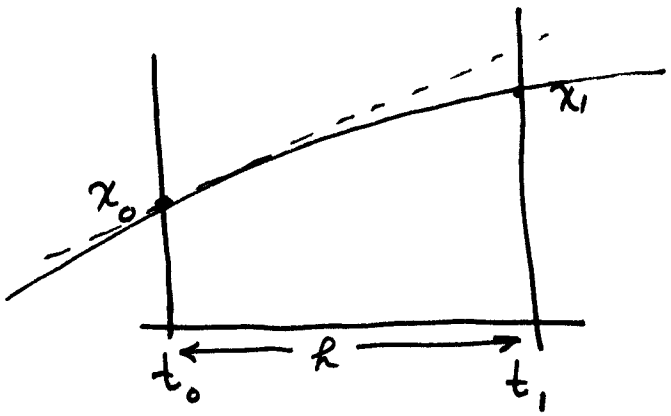
Euler's Method ("point-slope")

Simplest, basis of others, but just a start

[Euler, Institutiones Calculi Integralis, 1768]

Starting point $\boxed{\dot{x} = f(x)}$

remember that x & f are n-vectors, but that doesn't affect our results



$$\dot{x}_0 = f(x_0) \approx \frac{x_1 - x_0}{h} \Rightarrow x_1 \approx x_0 + h f(x_0)$$

this is the iteration: $\boxed{x_1 := x_0 + h f(x_0)}$

Local (Single-Step) Error

Taylor's series again,

$$x_1 = x_0 + h f(x_0) + \frac{h^2}{2} f'(\xi) \quad \text{some } \xi \in [t_0, t_1]$$

$$\therefore \quad E_{\text{single-step}} = x_1 - [x_0 + h f(x_0)] = O(h^2) \quad \text{as } h \to 0$$

Global (Accumulated) Error

integrating from $t=0$ to T, we have T/h steps

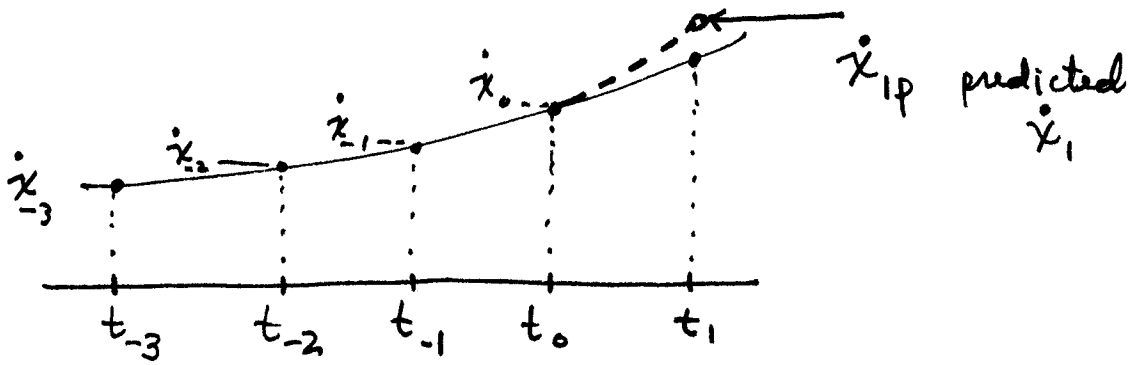
Heuristically, $\boxed{E_{\text{global}} = O(h)}$

(can be made rigorous, see J.R. Rice, Numerical Methods, Software, and Analysis, IMSL Reference Edition, McGraw-Hill, 1983)

Outline of Predictor-Corrector Idea

See [Acton 70] for details

1) Use past values to predict $\dot{x}_1$ from $\dot{x}_{-3}, \dot{x}_{-2}, \dot{x}_{-1}, \dot{x}_0$ (say):



2) integrate under fitted curve of $\dot{x}$ to estimate predicted x_{1p} at t_1

3) Correct derivative at t_1 via original equation:

$$\dot{x}_{1c} = f(x_{1p})$$

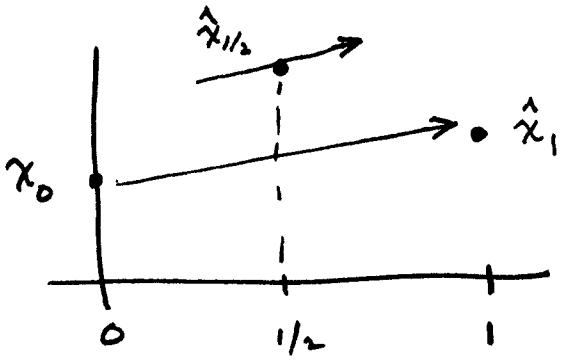
Can be iterated to correct more
has many variations

Numerical Recipes reports this has been superceded by Richardson extrapolation – in particular, Bulirsch-Stoer methods, which extrapolate to zero h. state-of-the-art?

We'll describe traditional, safe and easy to explain and program – Runge-Kutta.

Second-Order Runge-Kutta (Midpoint) Method

Idea:



① Use Euler with $h/2$ to find $\hat{x}_{1/2}$

② Use derivative $f(\hat{x}_{1/2})$ for a full step from x_0

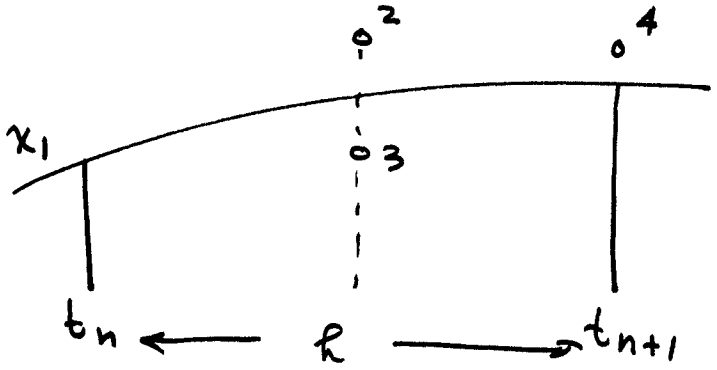
Local Error:

$$\begin{aligned}\hat{x}_1 &= x_0 + h f(\hat{x}_{1/2}) \\ &= x_0 + h\left[f(x_0) + \frac{h}{2}\dot{f}(x_0) + O(h^2)\right] \\ &= \underbrace{x_0 + h f(x_0) + \frac{h^2}{2}\dot{f}(x_0)}_{\text{first 3 terms of Taylor's series correct}} + O(h^3)\end{aligned}$$

$$\therefore \quad E_{local} = O(h^3)$$
$$E_{global} = O(h^2)$$

4th Order Runge-Kutta – Dependable Workhorse
- Classical
- Easy to program

f( ) evaluated four times per step



$$\boxed{\dot{x} = f(t, x)}$$

$$k_1 = h\, f(t_n, x_n)$$

$$k_2 = h\, f\left(t_n + \frac{h}{2},\ x_n + \frac{k_1}{2}\right)$$

$$k_3 = h\, f\left(t_n + \frac{h}{2},\ x_n + \frac{k_2}{2}\right)$$

$$k_4 = h\, f(t_n + h,\ x_n + k_3)$$

$$x_{n+1} = x_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(h^5)$$

{ local error $O(h^5)$
{ global error $O(h^4)$

See Num. Recipes also for adaptive step size methods plus more sophisticated stuff. this will do for Assgn. 3.

Stability

Simple example:

$$\dot{x} = -cx \qquad \text{one-dimensional}$$

Apply Euler's Method:

$$x_{n+1} = x_n + h\dot{x}_n = x_n - hcx_n$$

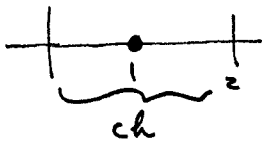
$$x_{n+1} = (1-ch)x_n$$

actual solution is $x_{n+1} = x_n e^{-ct} = x_n e^{-ch}$

converges if $ch \ll 1$, $h \ll 1/c$

In general will be stable if

$$|1-ch| < 1$$



otherwise diverges.

$$\boxed{h > 2/c}$$

In general, for explicit forward methods like Euler, Runge-Kutta, we require

$$h \ll 1/\lambda_{max}, \quad \text{where } \lambda_{max} = \text{largest eigenvalue}$$

Especially troublesome for "stiff" systems:

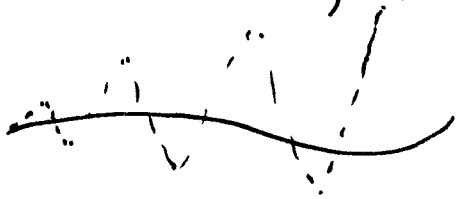
Say

$$x = e^{-t} + e^{-1000t} \leftarrow \text{solution}$$

$\lambda_{max} = 1000$, so we need $h \ll 1/1000$ for stability.

But after a short time, the e^{-1000t} term is negligable.

Easily leads to



Implicit Methods:

Simple example again

$$x_{n+1} = x_n + h\,\dot{x}_{n+1} \quad \text{"Backward Euler"}$$

(use derivative at right end)

$$= x_n + h(-c\,x_{n+1})$$

Solve:

$$x_{n+1} = \left(\frac{1}{1+ch}\right) x_n$$

$$|x_{n+1}| = \left|\frac{1}{1+ch}\right| |x_n| \leq |x_n|$$

In this simple case, $c>0$ corresponds to stable first-order system, & this is always stable!

---

also unconditionally stable for stable, linear, constant-coefficient eqns. of any order.

But these are only the simplest cases!

In nonlinear cases, solving the implicit eqn. is hard, and there is no guarantee of stability.

But these techniques are often a big improvement for "stiff" systems — see Num. Recipes for refinements of Runge-Kutta, etc. to implicit.

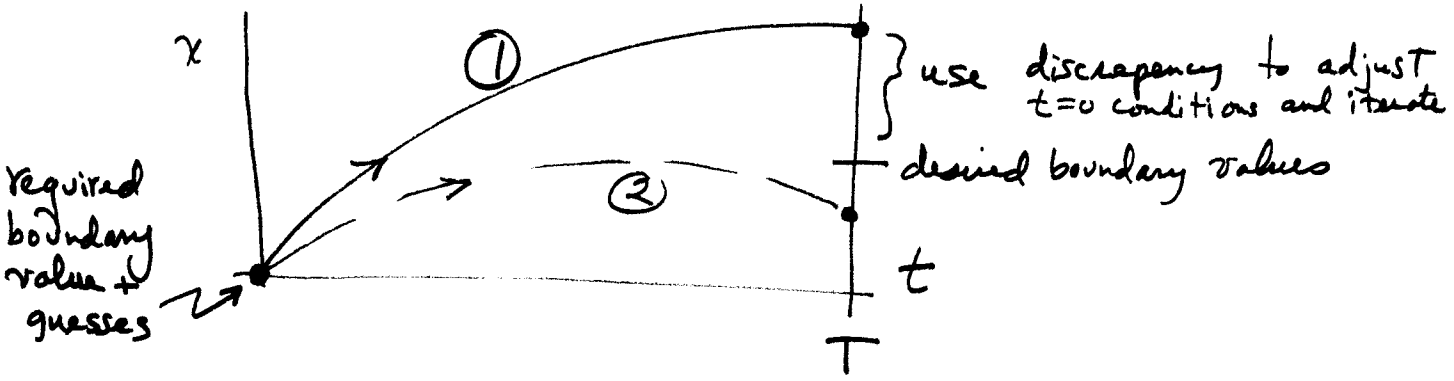
Briefly, Two-point Boundary Value Problems

Suppose we are given some conditions at $t=0$ and others at $t=T$ (final). What do we do?

A common approach is the

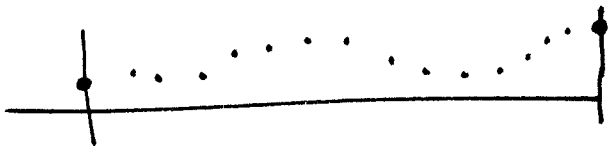
Shooting Method:

guess all conditions at $t=0$ and integrate:



Another method:

Relaxation: Start with (hopefully good) guess everywhere on grid:



adjust all values simultaneously

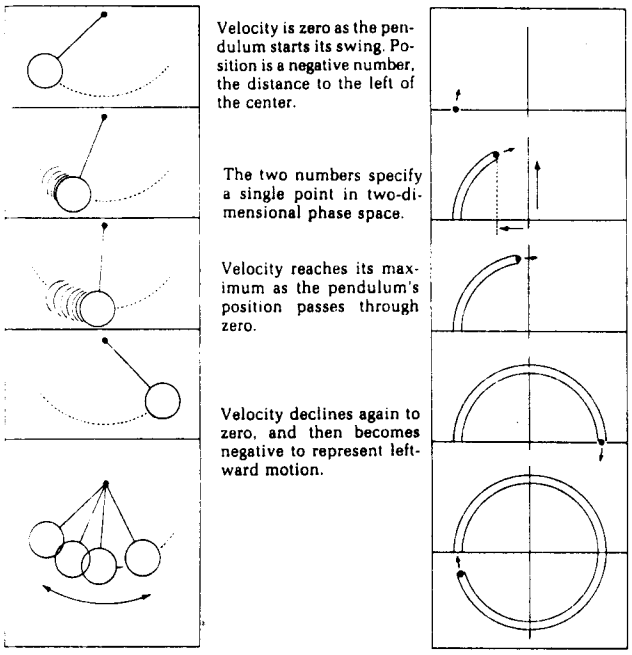
Sounds ridiculous, but can be good.

In summary, remember that solving differential equations is an Art as well as a science.

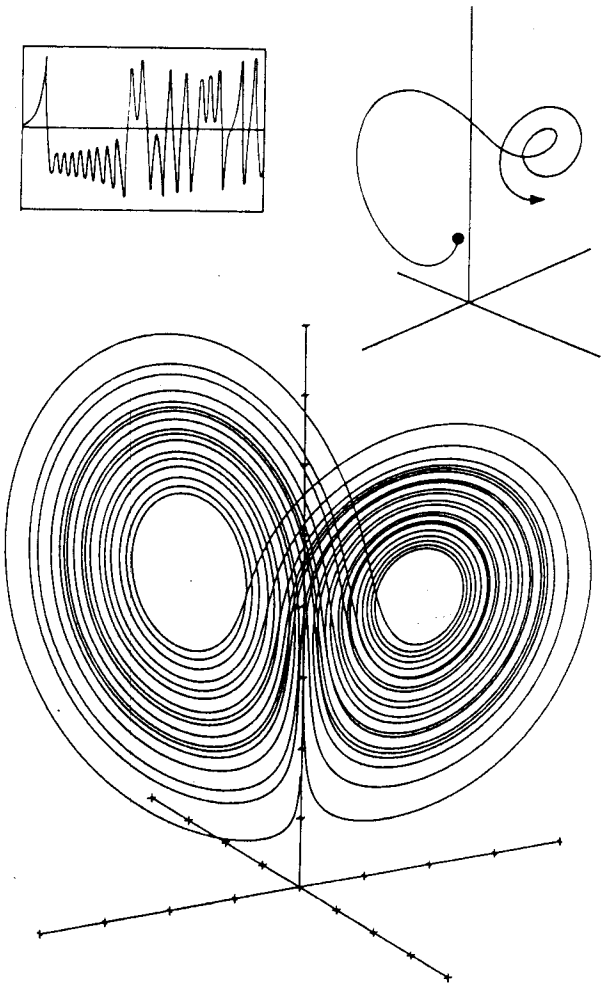
We have only touched the surface.

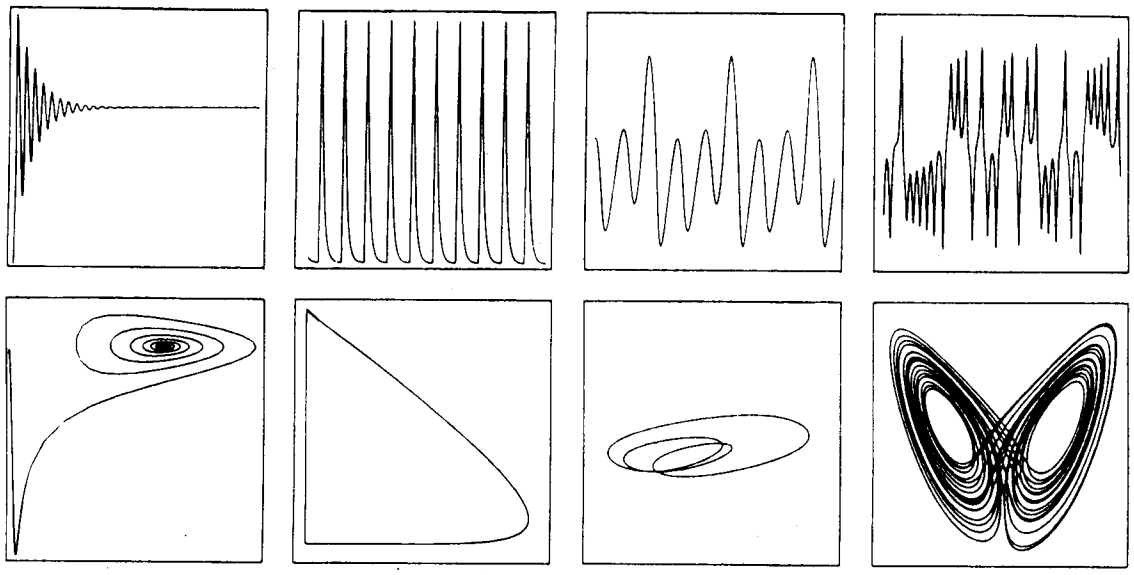
Phase-space Picture Gallery
[Gleick 87]

3.2.10



ANOTHER WAY TO SEE A PENDULUM. One point in phase space (*right*) contains all the information about the state of a dynamical system at any instant (*left*). For a simple pendulum, two numbers—velocity and position—are all you need to know.





MAKING PORTRAITS IN PHASE SPACE. Traditional time series (*above*) and trajectories in phase space (*below*) are two ways of displaying the same data and gaining a picture of a system's long-term behavior. The first system (*left*) converges on a steady state—a point in phase space. The second repeats itself periodically, forming a cyclical orbit. The third repeats itself in a more complex waltz rhythm, a cycle with "period three." The fourth is chaotic.