Math 2280-001
Week 4: Jan 30-Feb 3, sections 2.3, 2.4-2.6

Mon Jan 30
We'll use last week's notes to discuss section 2.3, improved velocity models

- Use Friday notes
- extra time: check out LCB 115
  → make sure you can log on
  → Maple?

Wed Feb 1
2.4-2.6 numerical methods for first order differential equations. We will probably meet in LCB 115 (TBA) to work through these notes, in Maple.

http://www.math.utah.edu/~korevaar/2280spring17/week4.mw

.....

Math Department login for students is as follows:

login name: Suppose your name is Public, John Q. Then your login name is

c-pcjq

(If several UU math students have the same initials as you, then you may actually be signed up as one of c-pcjq1, c-pcjq2, c-pcjq3 or c-pcjq4.)

password: If your UID is u***4986 then your password is

pcjq4986

(unless you've changed it). Even if you had to add a number to your login name, your password will only have the four letters part of your login name.

To open this document from Maple, go to our Math 2280 lecture page, then to the link above. Use the File/Open URL option inside Maple, and copy and paste the URL into the dialog box.

save a copy of this file to your home directory (using whatever name seems appropriate).

.....

In this handout we will study numerical methods for approximating solutions to first order differential equations. Later in the course we will see how higher order differential equations can be converted into first order systems of differential equations. It turns out that there is a natural way to generalize what we do now in the context of a single first order differential equations, to systems of first order differential equations. So understanding this material will be an important step in understanding numerical solutions to higher order differential equations and to systems of differential equations.

We will be working through material from sections 2.4-2.6 of the text.

Euler's Method:

The most basic method of approximating solutions to differential equations is called Euler's method, after the 1700's mathematician who first formulated it. Consider the initial value problem

$$\frac{dy}{dx} = f(x, y)$$
$$y(x_0) = y_0.$$

We make repeated use of the familiar approximation

$$\frac{dy}{dx} \approx \frac{\Delta y}{\Delta x}$$

with fixed "step size" $\Delta x := h$ at our discretion (the smaller the better, probably, if we want to be accurate). We consider approximating the graph of the solution to the IVP. Begin at the initial point (x_0, y_0). Let

$$x_1 = x_0 + h$$

To get the Euler approximation y_1 to the exact value $y(x_1)$ use the rate of change $f(x_0, y_0)$ at your initial point (x_0, y_0), i.e. $\Delta y \approx f(x_0, y_0)\Delta x$, so

$$y_1 = y_0 + f(x_0, y_0)h.$$

In general, if we've approximated (x_j, y_j) we set

$$x_{j+1} = x_j + h$$
$$y_{j+1} = y_j + f(x_j, y_j)h.$$

We could also approximate in the $-x$ direction from the initial point (x_0, y_0), by defining e.g.

$$x_{-1} = x_0 - h$$
$$y_{-1} = y_0 - hf(x_0, y_0)$$

and more generally via

$$x_{-j-1} = x_{-j} - h$$
$$y_{-j-1} = y_{-j} - hf(x_{-j}, y_{-j})$$

...etc.

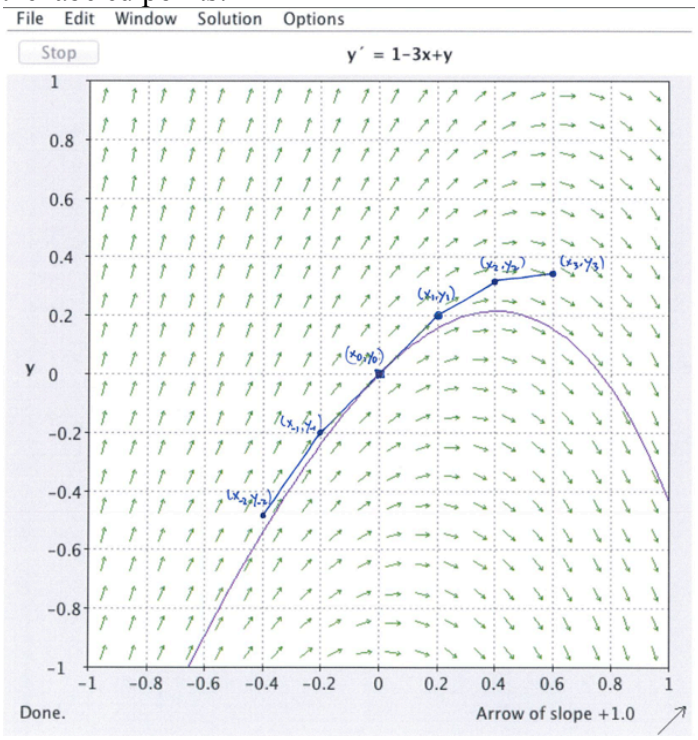
.

Below is a graphical representation of the Euler method, illustrated for the IVP

$$y'(x) = 1 - 3x + y$$
$$y(0) = 0$$

with step size $h = 0.2$.

Exercise 1 Find the numerical values for several of the labeled points.



As a running example in the remainder of these notes we will use one of our favorite IVP's from the time of Calculus, namely the initial value problem for $y(x)$,

$$y'(x) = y$$
$$y(0) = 1.$$

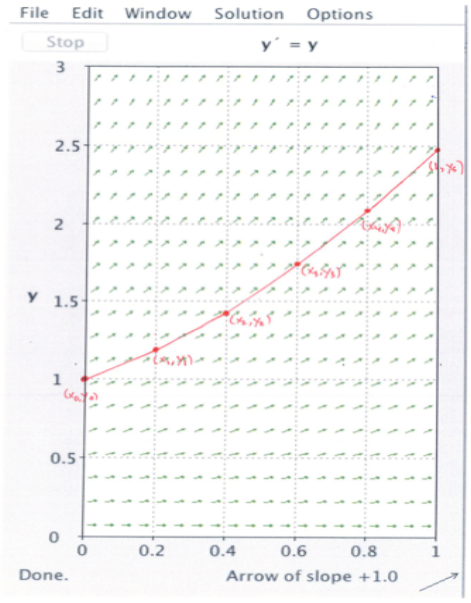
We know that $y = e^x$ is the solution.

Exercise 2 Work out by hand the approximate solution to the IVP above on the interval $[0, 1]$, with $n = 5$ subdivisions and $h = 0.2$, using Euler's method. This table might help organize the arithmetic. In this differential equation the slope function is $f(x, y) = y$ so is especially easy to compute.

step i	x_i	y_i	$k = f(x_i, y_i)$	$\Delta x = h$	$\Delta y = h \cdot k$	$x_i + \Delta x$	$y_i + \Delta y$
0	0	1	1	0.2	0.2	0.2	1.2
1	0.2	1.2					
2							
3							
4							
5							

Euler Table

Your work should be consistent with the picture below.



Here is the automated computation for the previous exercise, and a graph comparing the approximate solution points to the actual solution graph:

Exercise 3 Enter the following commands (by putting your cursor in the command fields and hitting enter). Discuss what each command is doing, and ask for any needed clarification. The first collection of commands is initializing the data. The second set is running the Euler loop. If you forget to initialize the data first, you might run into trouble trying to run the second set. No matter what is written in the document, commands are not executed until the cursor is put into a command field, and the <enter> or <return> key is pressed.

Initialize:

```
> restart : #clear all memory, if you wish
> Digits := 6 : #use floating point arithmetic with 6 digits.  could use more if desired
> unassign( `x`, `y`); # in case you used the letters elsewhere
> f := (x, y)→y; # slope field function for the DE  y′(x)=y
                 # change for different DE's!!!
> x[0] := 0; y[0] := 1; #initial point
  h := 0.2; n := 5;   #step size and number of steps
> exactsol := x→e^x; #exact solution for the IVP  y'=y, y(0)=1
                     #change (or omit) for different IVP's!!!
>
```

Euler Loop

```
> for i from 0 to n do  #this is an iteration loop, with index "i" running from 0 to n
    print(i, x[i], y[i], exactsol(x[i]) );
        #print iteration step, current x,y, values, and exact solution value
    k := f(x[i], y[i]); #current slope function value
    x[i + 1] := x[i] + h;
    y[i + 1] := y[i] + h·k;
  end do: #how to end a for loop in Maple
```

Plot results:

```
> with(plots) :
  Eulerapprox := pointplot({seq([x[i], y[i]], i = 0 ..n)}) : #approximate soln points
   exactsolgraph := plot(exactsol(t), t = 0 ..1, `color` = `black`) :  #exact soln graph
      #used t because x has been defined to be something else
   display({Eulerapprox, exactsolgraph}, title = `approximate and exact solution graphs`);
```

Exercise 4: Why are your approximations too small in this case, compared to the exact solution?

It should be that as your step size h gets smaller, your approximations to the actual solution get better. This is true if your computer can do exact math (which it can't), but in practice you don't want to make the computer do too many computations because of problems with round-off error and computation time, so for example, choosing $h = .0000001$ would not be practical. But, trying $h = 0.01$ in our previous initial value problem should be instructive.

If we change the n-value to 100 and keep the other data the same we can rerun our experiment, copying, pasting, modifying previous work.

Initialize

```
> restart : #clear all memory, if you wish
> unassign( `x`, `y`); # in case you used the letters elsewhere
> Digits := 6 :
> f := (x, y) → y; # slope field function for the DE  y'(x)=y
                   # change for different DE's!!!
> x[0] := 0; y[0] := 1; #initial point
   h := 0.01; n := 100;   #step size and number of steps
> exactsol := x → e^x; #exact solution for the IVP  y'=y, y(0)=1
                       #change (or omit) for different IVP's!!!
>
```

Euler Loop (modified using an "if then" conditional clause to only print every 10 steps)

```
> for i from 0 to n do  #this is an iteration loop, with index "i" running from 0 to n
      if frac( i/10 ) = 0
          then print(i, x[i], y[i], exactsol(x[i]) );
      end if: #only print every tenth time
    k := f(x[i], y[i]); #current slope function value
    x[i + 1] := x[i] + h;
    y[i + 1] := y[i] + h·k;
  end do: #how to end a for loop in Maple
```

plot results:

```
> with(plots) :
  Eulerapprox := pointplot( {seq( [x[i], y[i]], i = 0 ..n ) }, color = red) :
   exactsolgraph := plot(exactsol(t), t = 0 ..1, `color` = `black`) :
      #used t because x was already used has been defined to be something else
   display( {Eulerapprox, exactsolgraph}, title = `approximate and exact solution graphs`);
```

Exercise 5: For this very special initial value problem

$$y'(x) = y$$
$$y(0) = 1$$

which has $y(x) = e^x$ as the solution, set up Euler on the x-interval $[0, 1]$, with n subdivisions, and step size $h = \frac{1}{n}$. Write down the resulting Euler estimate for $\exp(1) = e$. What is the limit of this estimate as $n \to \infty$? You learned this special limit in Calculus!

In more complicated differential equations it is a very serious issue to find relatively efficient ways of approximating solutions. An entire field of mathematics, ``numerical analysis" deals with such issues for a variety of mathematical problems. Our text explores improvements to Euler in sections 2.5 and 2.6, in particular it discusses improved Euler, and Runge Kutta. Runge Kutta-type codes are actually used in commerical numerical packages, e.g. in Maple and Matlab.

Let's summarize some highlights from 2.5-2.6.

Suppose we already knew the solution $y(x)$ to the initial value problem

$$y'(x) = f(x, y)$$
$$y(x_0) = y_0.$$

If we integrate the DE from x to $x + h$ and apply the Fundamental Theorem of Calculus, we get

$$y(x + h) - y(x) = \int_x^{x+h} f(t, y(t))\, dt, \text{ i.e.}$$

$$y(x + h) = y(x) + \int_x^{x+h} f(t, y(t))\, dt.$$

One problem with Euler is that we approximate this integral above by $h \cdot f(x, y(x))$, i.e. we use the value at the left-hand endpoint as our approximation of the integrand, on the entire interval from x to $x + h$. This causes errors that are larger than they need to be, and these errors accumulate as we move from subinterval to subinterval and as our approximate solution diverges from the actual solution. The improvements to Euler depend on better approximations to the integral. These are subtle, because we don't yet have an approximation for $y(t)$ when t is greater than x, so also not for the integrand $f(t, y(t))$ on the interval $[x, x + h]$.

Improved Euler uses an approximation to the Trapezoid Rule to compute the integral in the formula

$$y(x+h) = y(x) + \int_x^{x+h} f(t, y(t))\, dt .$$

Recall, the trapezoid rule to approximate the integral

$$\int_x^{x+h} f(t, y(t))\, dt$$

would be

$$\frac{1}{2} h \cdot (f(x, y(x)) + f(x+h, y(x+h))).$$

Since we don't know $y(x+h)$ we approximate its value with unimproved Euler, and substitute into the formula above. This leads to the improved Euler "pseudocode" for how to increment approximate solutions.

Improved Euler pseudocode:

$k_1 = f(x_j, y_j)$ #current slope

$k_2 = f(x_j + h, y_j + h\,k_1)$ #use unimproved Euler guess for $y(x_j + h)$ to estimate slope function when $x = x_j + h$

$k = \frac{1}{2}(k_1 + k_2)$ #average of two slopes

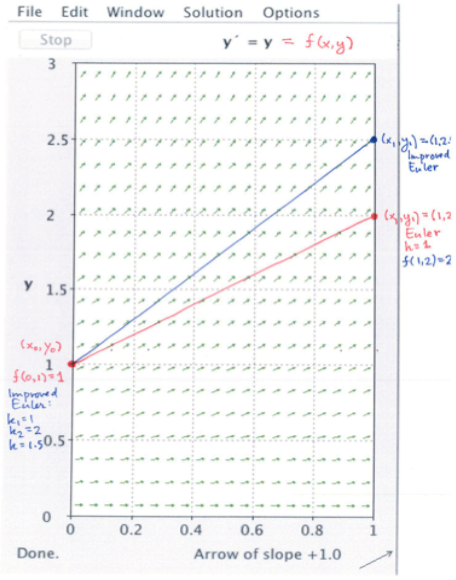
$x_{j+1} = x_j + h$ #increment x

$y_{j+1} = y_j + h\,k$ #increment y

Exercise 6 Contrast Euler and Improved Euler for our IVP

$$y'(x) = y$$
$$y(0) = 1$$

to estimate $y(1) \approx e$ with one step of size $h = 1$. Your computations should mirror the picture below. Note that with improved Euler you actually get a better estimate for e with ONE step of size 1 than you did with 5 steps of size $h = 0.2$ using unimproved Euler. (It's still not a very good estimate.)



In the same vein as ``improved Euler" we can use the Simpson approximation for the integral for Δy instead of the Trapezoid rule, and this leads to the Runge-Kutta method. You may or may not have talked about Simpson's Parabolic Rule for approximating definite integrals in Calculus. It is based on a quadratic approximation to the integrand g, whereas the Trapezoid rule is based on "linear" approximation.

Simpson's Rule:

Consider two numbers $x_0 < x_1$, with interval width $h = x_1 - x_0$ and interval midpoint $\underline{x} = \frac{1}{2}(x_0 + x_1)$.

If you fit a parabola $p(x)$ to the three points

$$\left(x_0, g\left(x_0\right)\right), \quad (\underline{x}, g(\underline{x})), \quad \left(x_1, g\left(x_1\right)\right)$$

then

$$\int_{x_0}^{x_1} p(t)\mathrm{d}t = \frac{h}{6}\left(g\left(x_0\right) + 4\cdot g(\underline{x}) + g\left(x_1\right)\right).$$

(You will check this fact in your homework this week!) This formula is the basis for Simpson's rule, which you can also review in your Calculus text or at Wikipedia. (Wikipedia also has a good entry on Runge-Kutta.) Applying the Simpson's rule approximation for our DE, and if we already knew the solution function $y(x)$, we would have

$$y(x+h) = y(x) + \int_x^{x+h} f(t, y(t))\,dt$$

$$\approx y(x) + \frac{h}{6}\cdot\left(f(x, y(x)) + 4f\left(x + \frac{h}{2}, y\left(x + \frac{h}{2}\right)\right) + f(x+h, y(x+h))\right).$$

However, we have the same issue as in Trapezoid - that we've only approximated up to $y(x)$ so far. Here's the pseudo-code for how Runge-Kutta takes care of this. (See also section 2.6 to the text.)

Runge-Kutta pseudocode:

$k_1 = f\left(x_j, y_j\right)$ # left endpoint slope

$k_2 = f\left(x_j + \frac{h}{2}, y_j + \frac{h}{2}\,k_1\right)$ # first midpoint slope estimate, using k_1 to increment y_j

$k_3 = f\left(x_j + \frac{h}{2}, y_j + \frac{h}{2}\,k_2\right)$ # second midpoint slope estimate using k_2 to increment y_j

$k_4 = f\left(x_j + h, y_j + h\,k_3\right)$ # right hand slope estimate, using k_3 to increment y_j

$k = \frac{1}{6}\left(k_1 + 2\,k_2 + 2\,k_3 + k_4\right)$ #weighted average of all four slope estimates, consistent with Simpson's rule

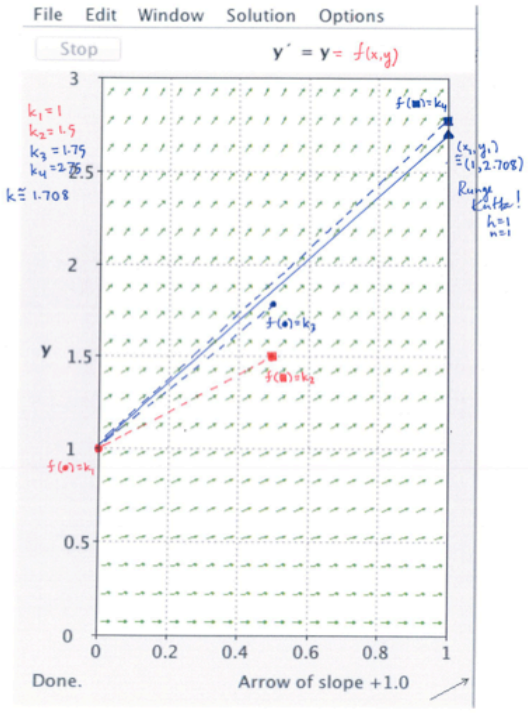
$x_{j+1} = x_j + h$ # increment x

$y_{j+1} = y_j + h\,k$ # increment y

Exercise 7 Contrast Euler and Improved Euler to Runge-Kutta for our IVP

$$y'(x) = y$$
$$y(0) = 1$$

to estimate $y(1) \approx$ e with one step of size $h = 1$. Your computations should mirror the picture below. Note that with Runge Kutta you actually get a better estimate for e with ONE step of size $h = 1$ than you did with 100 steps of size $h = 0.01$ using unimproved Euler!



Numerical experiments and homework

You can do the section 2.4-2.6 homework this week using the code snippets below. I'll illustrate by doing the "famous numbers" problem of estimating e.

Estimate e *by estimating* $y(1)$ for the solution to the IVP

$$y'(x) = y$$
$$y(0) = 1.$$

Apply Runge-Kutta with n = 10, 20, 40... subintervals, successively doubling the number of subintervals until you obtain the target number below - rounded to 9 decimal digits - twice in succession.

```
> evalf(e);
```

2.718281828459045 **(1)**

Initialize

```
> restart: # clear any memory from earlier work
> Digits := 15: # we need lots of digits for "famous numbers"
>
> unassign(`x`, `y`); # in case you used the letters elsewhere, and came back to this piece of code
> f := (x, y) -> y; # slope field function for our DE i.e. for y'(x)=f(x,y)
> x[0] := 0; y[0] := 1; #initial point
  h := 0.1; n := 10;   #step size and number of steps - first attempt for "famous number e"
```

Runge Kutta loop. WARNING: ONLY RUN THIS CODE AFTER INITIALIZING

```
> for i from 0 to n do #this is an iteration loop, with index "i" running from 0 to n
    print(i, x[i], y[i]); #print iteration step, current x,y, values, and exact solution value
    k1 := f(x[i], y[i]); #current slope function value
    k2 := f(x[i] + h/2, y[i] + h/2 * k1);
      # first estimate for slope at right endpoint of midpoint of subinterval
    k3 := f(x[i] + h/2, y[i] + h/2 * k2); #second estimate for midpoint slope
    k4 := f(x[i] + h, y[i] + h*k3); #estimate for right-hand slope
    k := (k1 + 2*k2 + 2*k3 + k4)/6; # Runge Kutta estimate for rate of change of y
    x[i + 1] := x[i] + h;
    y[i + 1] := y[i] + h*k;
  end do: #how to end a for loop in Maple
>
```

For other problems you may wish to use or modify the following Euler and improved Euler loops.

Euler loop: WARNING: ONLY RUN THIS CODE AFTER INITIALIZING

```
> for i from 0 to n do  #this is an iteration loop, with index "i" running from 0 to n
    print(i, x[i], y[i]);
        #print iteration step, current x,y, values, and exact solution value
    k := f(x[i], y[i]);  #current slope function value
    x[i + 1] := x[i] + h;
    y[i + 1] := y[i] + h·k;
  end do:  #how to end a for loop in Maple
>
```

Improved Euler loop: WARNING: ONLY RUN THIS CODE AFTER INITIALIZING

```
> for i from 0 to n do  #this is an iteration loop, with index "i" running from 0 to n
    print(i, x[i], y[i]);  #print iteration step, current x,y, values, and exact solution value
    k1 := f(x[i], y[i]);  #current slope function value
    k2 := f(x[i] + h, y[i] + h·k1); #estimate for slope at right endpoint of subinterval
    k := (k1 + k2)/2;  # improved Euler estimate
    x[i + 1] := x[i] + h;
    y[i + 1] := y[i] + h·k;
  end do:  #how to end a do loop in Maple
>
```

Math 2280-001
Fri Feb 3

• I'll post my version of filled in Wed. notes
• think about HW timing.

• Most likely, we did not finish Wednesday's notes. This will also be an opportunity to summarize the three numerical methods for solving DE's that we discussed: i.e. Euler, Improved Euler, and Runge Kutta, and for you to ask any questions you might have related to the 2.4-2.6 material. There is an application problem in this upcoming week's homework that will use numerical methods.

.....

• Then, begin Chapter 3, which is about higher order linear differential equations and applications.

3.1 Second order linear differential equations, and vector space theory connections to Math 2270:

Definition: A vector space is a collection of objects together with an "addition" operation "+", and a "scalar multiplication" operation, so that the rules below all hold.

(α) Whenever $f, g \in V$ then $f+g \in V$. (closure with respect to addition)
(β) Whenever $f \in V$ and $c \in \mathbb{R}$, then $c \cdot f \in V$. (closure with respect to scalar multiplication)

As well as:
(a) $f+g=g+f$ (commutative property) : at any x, $(f+g)(x) = f(x)+g(x) = g(x)+f(x) = (g+f)(x)$
(b) $f+(g+h)=(f+g)+h$ (associative property) ✓
(c) $\exists\ 0 \in V$ so that $f+0=f$ is always true. "0": $0(x)=0$
(d) $\forall f \in V\ \exists -f \in V$ so that $f+(-f)=0$ (additive inverses) : $(-f)(x) = -(f(x))$
(e) $c\cdot(f+g)=c\cdot f+c\cdot g$ (scalar multiplication distributes over vector addition)
(f) $(c_1+c_2)\cdot f = c_1\cdot f + c_2\cdot f$ (scalar addition distributes over scalar multiplication)
(g) $c_1\cdot(c_2\cdot f) = (c_1 c_2)\cdot f$ (associative property)
(h) $1\cdot f = f,\ (-1)\cdot f = -f,\ 0\cdot f = 0$ (these last two actually follow from the others).

Examples you've seen in Math 2270:
(1) $\mathbb{R}^m$, with the usual vector addition and scalar multiplication, defined component-wise $\vec{x} \in \mathbb{R}^n$
(2) subspaces W of $\mathbb{R}^m$, which satisfy (α),(β), and therefore automatically satisfy (a)-(h), because the vectors in W also lie in $\mathbb{R}^m$.

Maybe you've also seen ...
Exercise 1) In Chapter 3 we focus on the vector space
$$V = C(\mathbb{R}) := \{f : \mathbb{R} \to \mathbb{R} \text{ s.t. } f \text{ is a continuous function}\}$$
and its subspaces. Verify that the vector space axioms for linear combinations are satisfied for this space of functions. Recall that the function $f+g$ is defined by $(f+g)(x) := f(x)+g(x)$ and the scalar multiple $cf(x)$ is defined by $(cf)(x) := cf(x)$. What is the zero vector for functions?

or fixed subinterval of $\mathbb{R}$

e.g. in Calculus: $(f+g)' = f'+g'$
$(cf)' = cf'$

relation to $\vec{z} \in \mathbb{R}^n$:
$\vec{z} = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_n \end{bmatrix}$ scalar n components, one for each i, $1 \le i \le n$.
for f a function, get a "component" for each x

what is a linear combination of $f_1, f_2, \ldots f_n$ is any sum of scalar multiples
i.e. any $c_1f_1 + c_2f_2 + \cdots + c_nf_n$
where $c_1, c_2 \ldots c_n \in \mathbb{R}$

Recall that the vector space axioms are exactly the arithmetic rules we use to work with linear combination equations. In particular the following concepts are defined in any vector space V.

- the span of a finite collection of functions $\{f_1, f_2, \ldots f_n\}$. $= \text{span}\{f_1, f_2, \ldots f_n\}$ = set of all linear combos of $f_1 \ldots f_n$ $= \{c_1f_1 + c_2f_2 + \cdots + c_nf_n$ s.t. each $c_j \in \mathbb{R}\}$
- linear independence/dependence for a collection of functions $f_1, f_2, \ldots f_n$.
- subspaces of V : subset of V that is closed under $+$ & scalar. (subspace is a vector space)
- bases and dimension for finite dimensional subspaces. (The function space V itself is infinite dimensional, meaning that no finite collection of functions spans it.)

$\{f_1, f_2, \ldots f_n\}$ is linearly dependent : at least one f_j is a linear combo of the others
i.e. $c_1f_1 + c_2f_2 + \cdots + c_nf_n = 0$ for some choice $c_1, c_2 \ldots c_n$ where not all $c_j = 0$

independent (not dependent)
i.e. $c_1f_1 + c_2f_2 + \cdots + c_nf_n = 0 \iff c_1 = c_2 = \cdots = c_n = 0$.

a basis for W is a collection of vectors $\{f_1, f_2 \ldots f_n\}$ that span W and are linearly independent
# of vectors in any basis of W.

Definition: A second order linear differential equation for a function $y(x)$ is a differential equation that can be written in the form

$$A(x)y'' + B(x)y' + C(x)y = F(x) .$$

We search for solution functions $y(x)$ defined on some specified interval I of the form $a < x < b$, or (a, ∞), $(-\infty, a)$ or (usually) the entire real line $(-\infty, \infty)$. In this chapter we assume the function $A(x) \neq 0$ on I, and divide by it in order to rewrite the differential equation in the standard form

$$y'' + p(x)y' + q(x)y = f(x) .$$

contrast with 1st order linear $y' + p(x)y = q(x)$

Definition: The DE above is called homogeneous if the right hand side $f(x)$ is the zero function, $f(x) \equiv 0$. If f is not the zero function, the DE is called nonhomogeneous (or inhomogeneous).

One reason the DE above is called linear is that the "operator" L defined by

$$L(y) := y'' + p(x)y' + q(x)y$$

satisfies the so-called linearity properties

- (1) $L(y_1 + y_2) = L(y_1) + L(y_2)$
- (2) $L(cy) = cL(y), c \in \mathbb{R}$.

2270 $L(\vec{x}) = A\vec{x}$
$A(\vec{x} + \vec{y}) = A\vec{x} + A\vec{y}$
$A(c\vec{x}) = cA\vec{x}$
homog solns: $A\vec{x} = \vec{0}$.

(Recall that the matrix multiplication function $L(\underline{x}) := A\underline{x}$ satisfies the analogous properties. Any time we have have a transformation L satisfying (1),(2), we say it is a linear transformation.)

Exercise 2a) Check the linearity properties (1),(2) for the differential operator $L(y) := y'' + p(x)y' + q(x)y$.

$L(y_1 + y_2) = (y_1 + y_2)'' + p(x)(y_1 + y_2)' + q(x)(y_1 + y_2)$
$= y_1'' + y_2'' + p(x)(y_1' + y_2') + q(x)(y_1 + y_2)$
$= L(y_1) + L(y_2)$ ✓

$L(cy) = (cy)'' + p(x)(cy)' + q(x)(cy)$
$= cy'' + cp(x)y' + cq(x)y$
$= c(L(y))$

ex $L(y) = y'' + 3y' + 2y$

$L(x^2) = 2 + 3 \cdot 2x + 2x^2$ ← the fun $y(x) = x^2$
$= 2 + 6x + 2x^2$

$L(e^{-x}) = e^{-x} + 3(-e^{-x}) + 2e^{-x}$
$= 0$

$L(e^{-x}) = 0$ means e^{-x} solves homogeneous DE $y'' + 3y' + 2y = 0$

2b) Use these linearity properties to show that

Theorem 0 the solution space to the homogeneous second order linear DE

$$L(y) = y'' + p(x)y' + q(x)y = 0$$

is closed under addition and scalar multiplication, i.e. it is a subspace. Notice that this is the "same" proof one uses to show that the solution space to a homogeneous matrix equation $A\,\underline{\boldsymbol{x}} = \underline{\mathbf{0}}$ is a subspace.

Let $L(y_1) = 0$, $L(y_2) = 0$.
(i.e. Let y_1, y_2 be solns to the homogeneous DE.

then $L(y_1 + y_2) = L(y_1) + L(y_2) = 0 + 0 = 0$
so $y_1 + y_2$ is a homog. soln.

also $L(cy_1) = cL(y_1) = c \cdot 0 = 0$
so cy_1 is a homog. soln.

in 2270
"homogeneous soln space"
i.e. $\{\vec{x} \in \mathbb{R}^n \text{ s.t. } A\vec{x} = \vec{0}\}$
is a subspace:
Let $\vec{z}, \vec{w}$ be homogeneous solns, i.e. $A\vec{z} = \vec{0}$, $A\vec{w} = \vec{0}$
then $A(\vec{z} + \vec{w}) = A\vec{z} + A\vec{w} = \vec{0} + \vec{0} = \vec{0}$
$A(c\vec{z}) = cA\vec{z} = c\vec{0} = \vec{0}$

Exercise 3) As an example, find the solution space to the following homogeneous differential equation for $y(x)$

$$y'' + 2y' = 0$$

on the x-interval $-\infty < x < \infty$. Notice that the solution space is the span of two functions. Hint: This is really a first order DE for $v = y'$.

$v' + 2v = 0$ *

$e^{2x}(v' + 2v) = e^{2x} \cdot 0 = 0$

$\frac{d}{dx}(e^{2x}v) = 0$

$e^{2x}v = C$

$v = Ce^{-2x}$

$y'(x) = Ce^{-2x}$

$\Rightarrow y(x) = -\frac{C}{2}e^{-2x} + D$

$y(x) = c_1 e^{-2x} + c_2 \cdot 1$

soln space is span $\{e^{-2x}, 1\}$

Exercise 4) Use the linearity properties to show

Theorem 1 All solutions to the nonhomogeneous second order linear DE

$$y'' + p(x)y' + q(x)y = f(x)$$

are of the form $y = y_P + y_H$ where y_P is any single particular solution and y_H is some solution to the homogeneous DE. (y_H is called y_c, for complementary solution, in the text). Thus, if you can find a single particular solution to the nonhomogeneous DE, and all solutions to the homogeneous DE, you've actually found all solutions to the nonhomogeneous DE.

Theorem 2 (Existence-Uniqueness Theorem): Let $p(x), q(x), f(x)$ be specified continuous functions on the interval I, and let $x_0 \in I$. Then there is a unique solution $y(x)$ to the initial value problem

$$y'' + p(x)y' + q(x)y = f(x)$$
$$y(x_0) = b_0$$
$$y'(x_0) = b_1$$

and $y(x)$ exists and is twice continuously differentiable on the entire interval I.

Exercise 5) Verify Theorems 1 and 2 for the interval $I = (-\infty, \infty)$ and the IVP

$$y'' + 2y' = 3$$
$$y(0) = b_0$$
$$y'(0) = b_1$$

Unlike in the previous example, and unlike what was true for the first order linear differential equation

$$y' + p(x)y = q(x)$$

there is not a clever integrating factor formula that will always work to find the general solution of the second order linear differential equation

$$y'' + p(x)y' + q(x)y = f(x).$$

Rather, we will usually resort to vector space theory and algorithms based on clever guessing to solve these differential equations. It will help to know

Theorem 3: The solution space to the second order homogeneous linear differential equation

$$y'' + p(x)y' + q(x)y = 0$$

is 2-dimensional.

This Theorem is illustrated in Exercise 2 that we completed earlier. Theorem 3 and the techniques we'll actually be using going forward are illustrated by

Exercise 6) Consider the homogeneous linear DE for $y(x)$

$$y'' - 2y' - 3y = 0$$

6a) Find two exponential functions $y_1(x) = e^{rx}, y_2(x) = e^{\rho x}$ that solve this DE. Deduce that arbitrary linear combinations of y_1, y_2 also solve the DE.

6b) Show that every IVP

$$y'' - 2y' - 3y = 0$$
$$y(0) = b_0$$
$$y'(0) = b_1$$

can be solved with a unique linear combination $y(x) = c_1 y_1(x) + c_2 y_2(x)$.

6c) Use your work from part b to explain why the solution space is two-dimensional.

6d) Now consider the nonhomogeneous DE

$$y'' - 2y' - 3y = 9$$

Notice that $y_P(x) = -3$ is a particular solution. Use this information and superposition (linearity) to find the solution to the initial value problem

$$y'' - 2y' - 3y = 9$$
$$y(0) = 6$$
$$y'(0) = -2.$$