MAT 305: Mathematical Computing

Interactive worksheets in Sage

John Perry

University of Southern Mississippi

Spring 2017

Outline

1. Interactive worksheets
2. Interactive objects
3. Detailed example
4. Summary

You should be in worksheet mode to repeat the examples.

Outline

1. Interactive worksheets
2. Interactive objects
3. Detailed example
4. Summary

Interactive worksheets?

An *interactive worksheet* allows a user to visualize and manipulate concepts in a hands-on fashion.

- buttons, sliders, checkboxes
- graphics updated immediately or on demand

Creating interactive worksheets

“Function decorator”: `@interact`

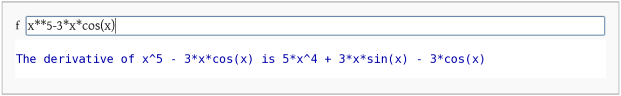
- Place immediately before definition of function
- Formal argument list consists of interact objects
    - input box
    - slider
    - checkbox
    - dropdown menu
    - buttons
    - color selector

Example

```
sage:  @interact
       def i_deriv(f=input_box(label='$f$')):
         if (f != None):
           print 'The derivative of ', f,
                 'is', diff(f)
```

Example

```
sage:  @interact
       def i_deriv(f=input_box(label='$f$')):
         if (f != None):
           print 'The derivative of ', f,
                 'is', diff(f)
```



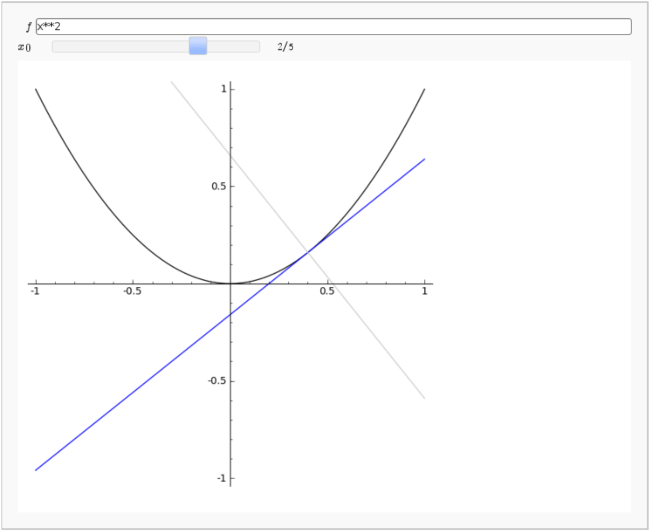
f x**5-3*x*cos(x)

```
The derivative of x^5 - 3*x*cos(x) is 5*x^4 + 3*x*sin(x) - 3*cos(x)
```

Something more visual

```
sage:  xmin, xmax = -1, 1
sage:  @interact
       def i_tan_norm(f=input_box(label=’$f$’),
           x0=slider(xmin,xmax,label=’$x_0$’,
           step_size=1/10,default=0)):
         if f != None and f != ’’:
           y0 = f(x=x0)
           mtan = (diff(f))(x=x0)
           mnorm = -1/mtan
           fplot = plot(f,xmin,xmax,color=’black’)
           tan_plot = plot(mtan*(x-x0)+y0,xmin,xmax)
           norm_plot = plot(mnorm*(x-x0)+y0,xmin,
                            xmax,color=(0.8,0.8,0.8))
           show(fplot+tan_plot+norm_plot,ymin=-1,
                ymax=1,aspect_ratio=1)
```

…the result



f x**2

$x0$ 2/5

Outline

1. Interactive worksheets
2. Interactive objects
3. Detailed example
4. Summary

Usage

- argument to interactive function
- *id* = *object*(*options*) where
  - *id* is an argument for the value of the object
  - *object* is one of the object commands given below
  - *options* specify object's properties
    - two options common to all objects

Command options for all objects

- `label` = *label*
  a string that labels the object
  - limited LaTeX
    (`latex()` command can be useful!)
  - compare `label=’x_0’`, `label=’$x_0$’`
- `default` = *value*
  the default value of the object, if any

The `input_box()` command

`input_box(`*options*`)` where *options* include

- `width`: width of box (# letters)

User enters text (function, number, etc.)

Example

```
f = input_box(label='$f$',default=x*cos(x),width=10)
```

The `slider()` command

`slider(`*options*`)` where *options* include

- continuous slider?
  - `start`: minimum value of slider
  - `stop`: maximum value of slider
- discrete slider? two ways
  1. list of values: `start`, no `stop`
  2. range and step size: `start`, `stop`, `step`

User slides knob across line to select value

Example

```
x0 = slider(label='$x_0$',vmin=-1,vmax=1,
            default=0,step_size=1/10)
```

The `checkbox()` command

`checkbox(`*options*`)`

- User sets boolean (on/off or `True`/`False`) value

Example

```
show_tangent = checkbox(label='show tangent',
                         default=True)
```

Choosers

`selector(`*options*`)` where *options* include

- `values`: list of values or (*value*,*label*) pairs
- `buttons`: draw buttons, not a drop-down menu, if `True`
- `nrows`, `ncols`: number of rows or columns of buttons
- `width`: set all buttons to same length (in characters)

User chooses one of several options

Example

```
function = selector(values=[’normal line’,
                          ’tangent line’,
                          ’both’,’neither’])
```

Color selector

`Color(`*color definition*`)` where

- *color definition* is
  - a recognized name for a color
  - an rgb triplet
  - a hex string (don't worry about this one unless you already know what I mean)
- "common" options do not work with this object

User manipulates color using string, circle, box

Example

```
col = Color(0,0,1)
```

Outline

1. Interactive worksheets
2. Interactive objects
3. Detailed example
4. Summary

Example problem

Problem

Given f and c, plot f, its derivative df, and a line tangent to f at $x = c$.

Reuse old code

We already know how to compute line tangent to a function: (don’t retype)

```
sage:  def tangent_line(f, a, x=x):
         # redefine f
         f(x) = x
         # point-slope form of a line
         b = f(x=a)
         df(x) = diff(f,x)
         m = df(a)
         result = m*(x - a) + b
         return result
```

Reuse old code

We already know how to compute line tangent to a function: (don't retype)

```
sage:  def tangent_line(f, a, x=x):
         # redefine f
         f(x) = x
         # point-slope form of a line
         b = f(x=a)
         df(x) = diff(f,x)
         m = df(a)
         result = m*(x - a) + b
         return result
```

Saved in `calc_utils.py` , so:

```
sage:  %attach calc_utils.sage
```

Interactive function

```
inputs
    $f$, a function
    $c \in \mathbb{R}$
do
    plot $f$
    plot $f'$
    plot line tangent to $f$ at $x = c$
```

Interactive function

```
inputs
    $f$, a function
    $c \in \mathbb{R}$
do
    plot $f$
    plot $f'$
    plot line tangent to $f$ at $x = c$
```

…over what interval? **refine pseudocode**

Interactive function

```
inputs
    $f$, a function
    $a, b \in \mathbb{R}$
    $t \in [0, 1]$
do
    plot $f$ on $[a, b]$
    plot $f'$ on $[a, b]$
    let $c = t(b - a)$
    plot on $[a, b]$ line tangent to $f$ at $x = c$
```

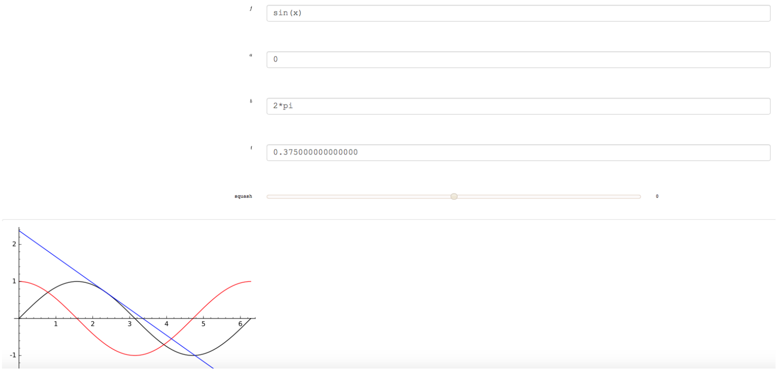
Interactive function

```
inputs
    $f$, a function
    $a, b \in \mathbb{R}$
    $t \in [0, 1]$
do
    plot $f$ on $[a, b]$
    plot $f'$ on $[a, b]$
    let $c = t(b - a)$
    plot on $[a, b]$ line tangent to $f$ at $x = c$
(parametrized $c$ on $[a, b]$ using $t$)
```

Sage code

```
sage: @interact
      def plot_f_df_tline(
            f=input_box(default=sin(x), label=’$f$’),
            a=input_box(default=0, label=’$a$’),
            b=input_box(default=2*pi, label=’$b$’),
            t=slider(0, 1, default=0.375, label=’$t$’),
            w=slider(-19, 19, 1, default=0, label=’squash’)):
          p = plot(f, a, b, color=’black’)
          df = diff(f)
          p = p + plot(df, a, b, color=’red’)
          # t tells us how far to move along [a,b]
          # 0 means a; 1 means b; other values proportional
          c = a + t*(b - a)
          p = p + plot(tangent_line(f, c), a, b)
          show(p, aspect_ratio=(20-w)/20)
```

Result



f sin(x)

a 0

b 2*pi

t 0.375000000000000

squash 0

Outline

1. Interactive worksheets

2. Interactive objects

3. Detailed example

4. Summary

Summary

- Interactive worksheets help user visualize, manipulate concepts
- Use `@interact` function decorator
- Several easy-to-define interface objects
- Break functions into parts
  - easy to read
  - easy to reuse
  - easy to change