Feb 20, 2024

We discussed this:

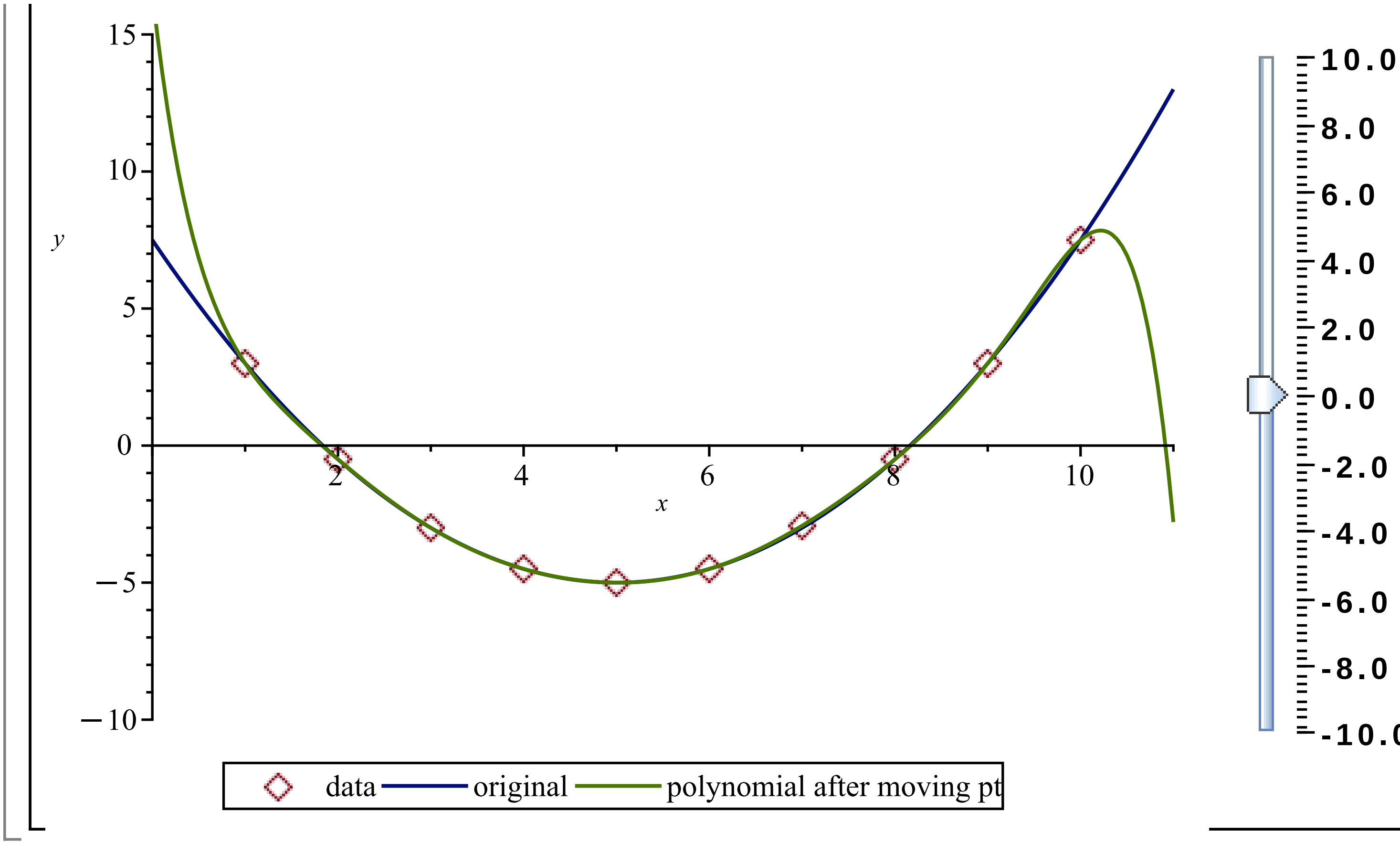
startup code

```
> base := x→(x − 5)^2/2 − 5 :  # define  my polynomial
  pdata :=proc(b)  # generate some points including [7, base(7)+b]
     local  i, data;
     data := [seq([i, base(i)], i=1 ..10)];  # generate base data
     data[7, 2] := data[7, 2] + b;    # push the 7th entry by b
     return(data);
   end:
   f := b→CurveFitting[PolynomialInterpolation](pdata(b), x) :  # fit a curve
  pl := b→plot([pdata(b), f(0), f(b)], x=0 ..11, y=−10 ..15,        #make a plot
           style=[point, line$2], symbolsize=20,
          legend=["data", "original", "polynomial after moving pt"]) :
```

my interactive thingy is here.

polyfit is

$-0.0000185\,x^{9} + 0.000890\,x^{8} - 0.0182\,x^{7} + 0.208\,x^{6} - 1.46\,x^{5} + 6.46\,x^{4} - 17.9\,x^{3} + 30.3\,x^{2} - 31.7\,x + 1$



> *pdata*(0);

$$\left[[1, 3], \left[2, -\frac{1}{2}\right], [3, -3], \left[4, -\frac{9}{2}\right], [5, -5], \left[6, -\frac{9}{2}\right], [7, -3], \left[8, -\frac{1}{2}\right], [9, 3], \left[10, \frac{15}{2}\right]\right] \tag{1}$$

> *pdata*(1)

$$\left[[1, 3], \left[2, -\frac{1}{2}\right], [3, -3], \left[4, -\frac{9}{2}\right], [5, -5], \left[6, -\frac{9}{2}\right], [7, -2], \left[8, -\frac{1}{2}\right], [9, 3], \left[10, \frac{15}{2}\right]\right] \tag{2}$$

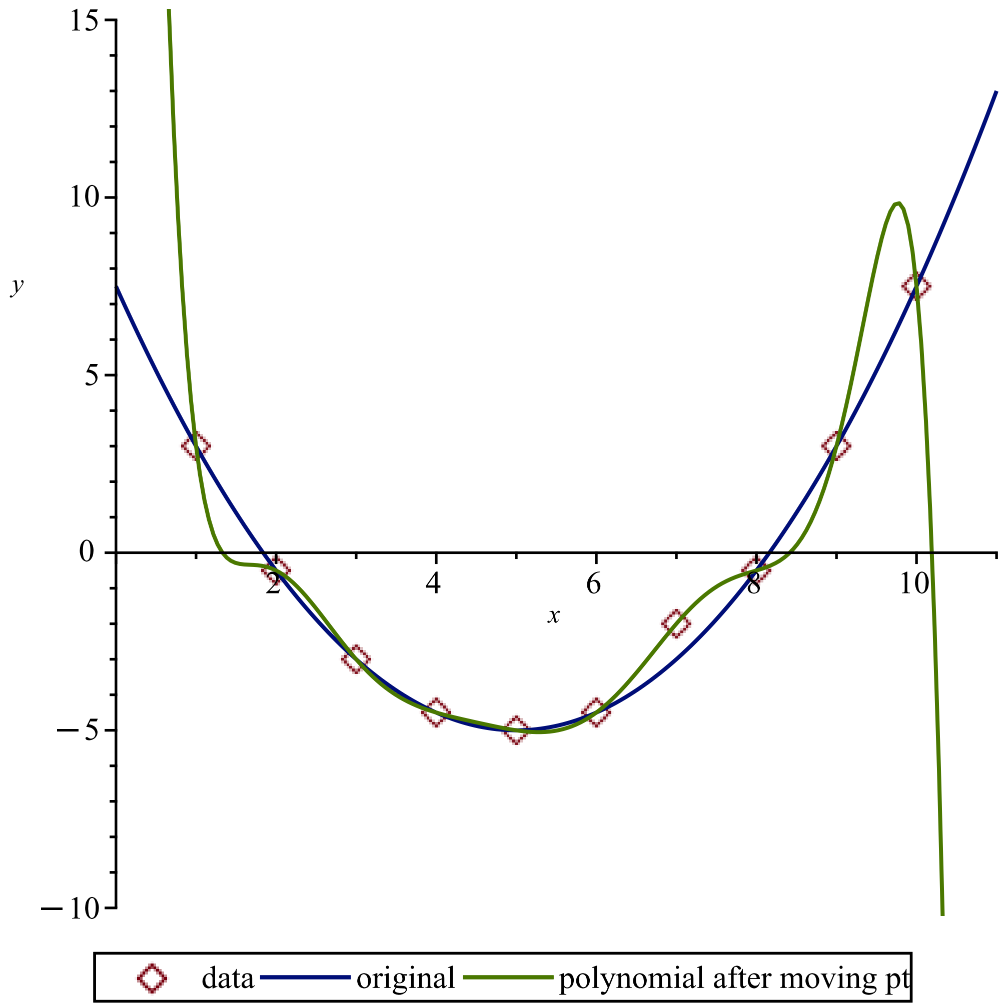
> *f*(0);

$$\frac{1}{2} x^2 - 5 x + \frac{15}{2} \tag{3}$$

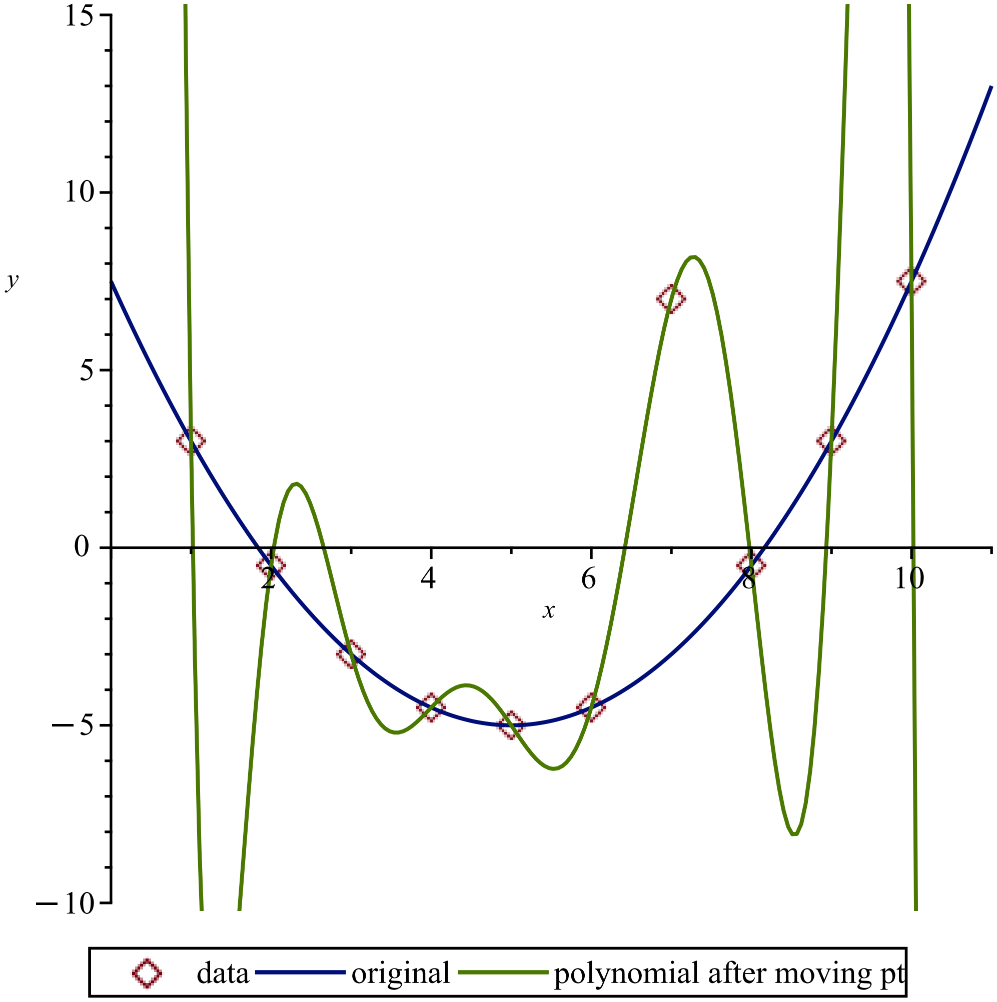
> *f*(1)

$$-\frac{1}{4320} x^9 + \frac{1}{90} x^8 - \frac{41}{180} x^7 + \frac{1877}{720} x^6 - \frac{8771}{480} x^5 + \frac{58247}{720} x^4 - \frac{242639}{1080} x^3 + \frac{22469}{60} x^2 - \frac{1018}{3} x + \frac{255}{2} \tag{4}$$

> *pl*(1)


15
10
5
0
−5
−10
y
x
2
4
6
8
10
◇ data
original
polynomial after moving pt

```
> pl(10)
```



What does the local i mean in the definition of pdata do?
side trip...

```
>  joe ≔ 3;
                                   joe ≔ 3                                        (5)
```

```
>  seq(i^2, i = 2 ..5)
                                 4, 9, 16, 25                                     (6)
```

```
>  for i from 2 to 5 do
      print(i, i^2);
   od;
                                    2, 4
                                    3, 9
                                    4, 16
                                    5, 25                                         (7)
```

```
>  squares ≔ [cat, dog, elephant, yommama, 64];
                   squares ≔ [cat, dog, elephant, yommama, 64]                    (8)
```

```
>
   for i from 2 to 5 do
```

```
    print(i, i^2);
    squares[i] := i^2;
od;
                              2, 4
              squares := [cat, 4, elephant, yommama, 64]
                              3, 9
                 squares := [cat, 4, 9, yommama, 64]
                              4, 16
                    squares := [cat, 4, 9, 16, 64]
                              5, 25
                    squares := [cat, 4, 9, 16, 25]                    (9)
```

```
> squares[3]
                                9                                      (10)
```

```
> for i from 2 to 5 do
    print(i, i^2) :
    squares[i] := i^2 :
  od:
                              2, 4
                              3, 9
                              4, 16
                              5, 25                                    (11)
```

```
> i := 3;
                              i := 3                                   (12)
```

```
> g :=proc(x)
    i := x^2;
    print("i is ", i);
   end;
Warning, (in g) `i` is implicitly declared local
        g := proc(x) local i; i := x^2; print("i is ", i) end proc     (13)
```

```
> g(3)
                            "i is ", 9                                 (14)
```

```
> i
                                3                                      (15)
```

Inside the procedure g, i is not the value outside. to make this clear, we can tell our program that i is local or not.

```
> h :=proc(x)
    local i;
    global j;
    i := x^2;
    j := i;
    print("i is ", i, "j is ", j);
   end;
```

```
        h := proc(x)  local i;  global j;  i := x^2; j := i; print("i is ", i, "j is ", j)  end proc        (16)
> j := 27;
                                        j := 27                                                        (17)
> i, j;
                                         3, 27                                                         (18)
> h(2);
                                  "i is ", 4, "j is ", 4                                               (19)
> i, j
                                          3, 4                                                         (20)
> h :=proc(x)
    local i;
    global j;
    i := x^2;
    j := i;
    print("i is ", i, "j is ", j);
    return(x + j);
  end;
h := proc(x)  local i;  global j;  i := x^2; j := i; print("i is ", i, "j is ", j);  return x + j  end proc (21)
> h(1)
                                  "i is ", 1, "j is ", 1
                                            2                                                          (22)
> h(4)
                                 "i is ", 16, "j is ", 16
                                           20                                                          (23)
> pdata :=proc(b)  # generate some points including [7, base(7)+b]
    local  i, data;
    data := [seq([i, fase(i)], i=1 ..10)];  # generate base data
    data[7, 2] := data[7, 2] + b;    # push the 7th entry by b
    return(data);
  end:
>  pdata(0)
[[1, fase(1)], [2, fase(2)], [3, fase(3)], [4, fase(4)], [5, fase(5)], [6, fase(6)], [7, fase(7)],     (24)
    [8, fase(8)], [9, fase(9)], [10, fase(10)]]
> base(x);
```

$$\frac{(x-5)^2}{2} - 5 \tag{25}$$

```
> pdata :=proc(b)  # generate some points including [7, base(7)+b]
    local  i, data;
    data := [seq([i, base(i)], i=1 ..10)];  # generate base data
    data[7, 2] := data[7, 2] + b;    # push the 7th entry by b
    return(data);
  end:
> mystuff := pdata(0);
```

$$mystuff := \left[[1, 3], \left[2, -\frac{1}{2}\right], [3, -3], \left[4, -\frac{9}{2}\right], [5, -5], \left[6, -\frac{9}{2}\right], [7, -3], \left[8, -\frac{1}{2}\right], [9, 3], \left[10, \frac{15}{2}\right]\right] \qquad \textbf{(26)}$$

```
> mystuff[7]
```

$$[7, -3] \qquad \textbf{(27)}$$

```
> mystuff[7][2]
```

$$-3 \qquad \textbf{(28)}$$

```
> gdata :=proc(b)  # generate some points including [7, base(7)+b]
     local  i;
     global data;
     data := [seq([i, base(i)], i=1..10)];  # generate base data
     data[7, 2] := data[7, 2] + b;    # push the 7th entry by b
     return(data);
   end:
```

```
> data := somestuuffffffff
```

$$data := somestuuffffffff \qquad \textbf{(29)}$$

```
> result := gdata(5)
```

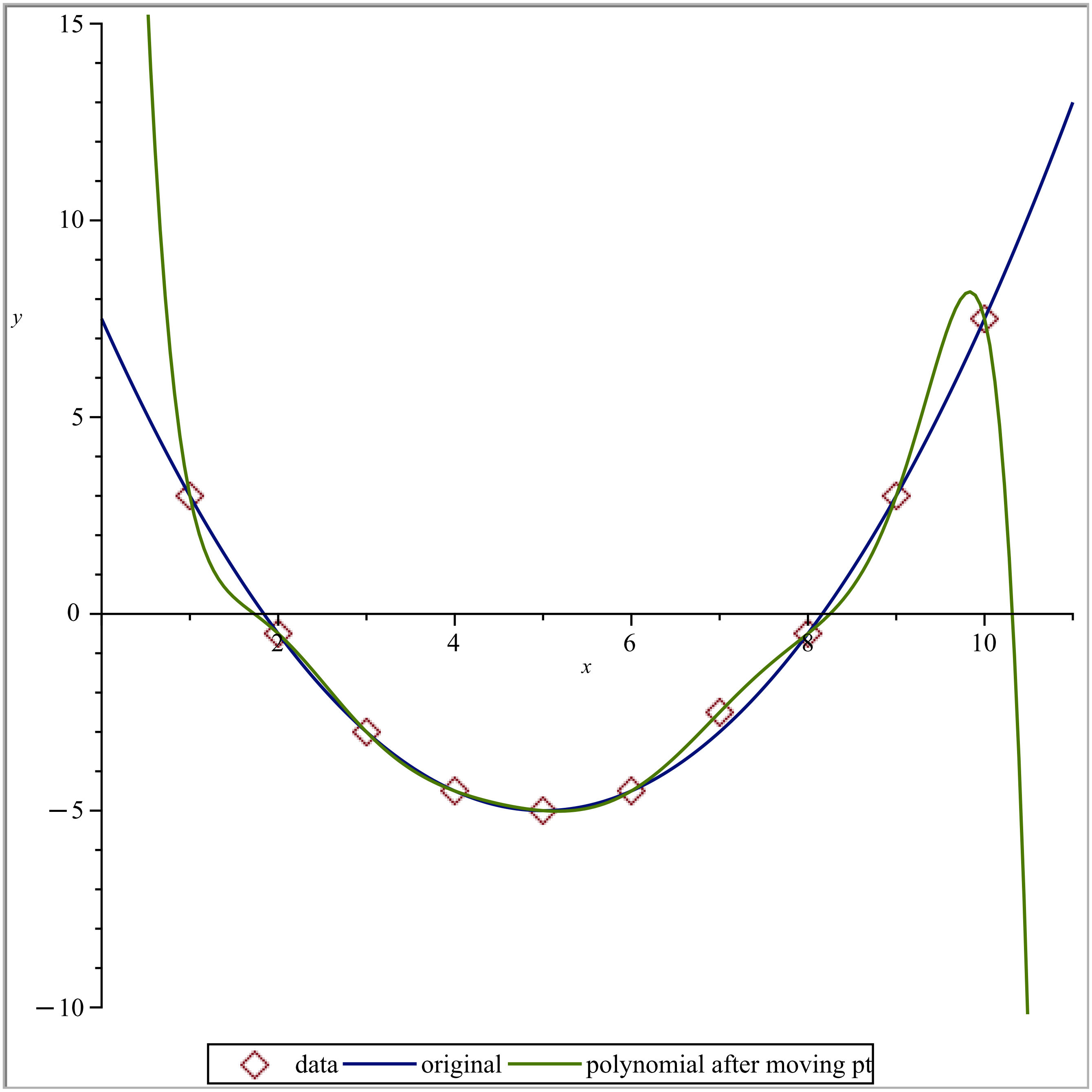
$$result := \left[[1, 3], \left[2, -\frac{1}{2}\right], [3, -3], \left[4, -\frac{9}{2}\right], [5, -5], \left[6, -\frac{9}{2}\right], [7, 2], \left[8, -\frac{1}{2}\right], [9, 3], \left[10, \frac{15}{2}\right]\right] \qquad \textbf{(30)}$$

```
> data
```

$$\left[[1, 3], \left[2, -\frac{1}{2}\right], [3, -3], \left[4, -\frac{9}{2}\right], [5, -5], \left[6, -\frac{9}{2}\right], [7, 2], \left[8, -\frac{1}{2}\right], [9, 3], \left[10, \frac{15}{2}\right]\right] \qquad \textbf{(31)}$$

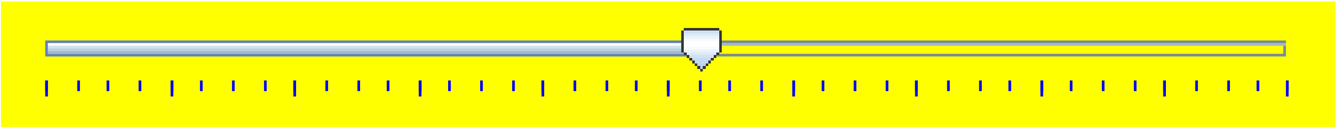
```
>
>
>
```

in the components menu on the side <----   are some things. I used the plot (looks like a plot of a surface)   click on the empty box, I get stuff in the right side



select a slider, see what

>



right click on the slider, Edit Value Changed Code, change it to update the plot.

```
> # in the value changed code, it says
  use DocumentTools in
  Do( %Plot1 = pl(%Slider1) );
```

```
    end use;
>
>
>
>
>
>
```