Subject: Re: Yet another user with poly_fit problems
Posted by suicidaleggroll on Mon, 30 Sep 2013 20:18:03 GMT
View Forum Message <> Reply to Message

On Monday, September 30, 2013 2:09:10 PM UTC-6, David Fanning wrote:
> Gus writes:
>
>
>
>> I've read a few of the older posts on this topic, but their solution didn't really help me solve the problem that I am currently having with the poly_fit function. The set of coefficients generated by the function (a 4th degree polynomial) produces some rather absurd results. Here is a short version of the problem I am having.
>
>>
>
>> X = [0.000000, 11.6667, 822.914, 3458.85, 27703.4, 133928.]
>
>> Y = [15.9000, 16.0000, 17.0000, 18.0000, 19.0000, 20.0000]
>
>>
>
>> C = poly_fit(X, Y, /double, yfit=D)
>
>>
>
>> IDL generates the following coefficients (for C)
>
>>
>
>> 15.940691
>
>> 0.0015355228
>
>> -3.0965110e-007
>
>> 1.1170193e-011
>
>> -6.6767399e-017
>
>
>
> This code doesn't seem to work for me:
>
>
>
> IDL> X = [0.000000, 11.6667, 822.914, 3458.85, 27703.4, 133928.]
>

> IDL> Y = [15.9000, 16.0000, 17.0000, 18.0000, 19.0000, 20.0000]
>
> IDL> C = poly_fit(X, Y, /double, yfit=D)
>
> % Compiled module: POLY_FIT.
>
> % Variable is undefined: NDEGREE.
>
>
>
> Are you sure you are using the right POLY_FIT?
>
>
>
> Cheers,
>
>
>
> David
>
>
>
> --
>
> David Fanning, Ph.D.
>
> Fanning Software Consulting, Inc.
>
> Coyote's Guide to IDL Programming: http://www.idlcoyote.com/
>
> Sepore ma de ni thue. ("Perhaps thou speakest truth.")

He just missed the "4" in the call (for a 4th order polynomial).

Gus - actually Excel gives the EXACT same answer as IDL, which, as you said, is completely ridiculous. The problem is you're fitting a 4th order polynomial to 5 data points. Because of this, the solution will be mathematically perfect (R^2 = 1), because the solution is not overdetermined and no least squares fitting can be performed.

You need more points in order to generate a "valid" 4th order poly fit so the "fit" can actually do some good, rather than just reproduce your 5 values exactly (with god knows what in between).

I've run into this in the past, and in that application it was reasonable to linearly interpolate my 5 points to, say, 1000 points, and then perform the poly fit on that.

For example:
X = [0.000000, 11.6667, 822.914, 3458.85, 27703.4, 133928.]

```
Y = [15.9000, 16.0000, 17.0000, 18.0000, 19.0000, 20.0000]

newX = dindgen(1000)/999 * (max(X)-min(X)) + min(X)
newY = interpol(Y, X, newX)
C = poly_fit(newX, newY, 4, /double, yfit=D)
print, C
      17.356485
  0.00010074819
 -2.0171981e-09
  1.8082251e-14
 -5.6591322e-20
```