Subject: Re: MPFITFUN error -- only reading the first data value
Posted by graham kerr on Thu, 02 Apr 2015 10:02:05 GMT
View Forum Message <> Reply to Message

So after much staring at code I think I have found my (somewhat daft) mistake!

I think that in mpfitfun (and mpfit & mpfitexpr), the function that you specify ('myfunction') must have the independent variable set as 'x' and the dependent variable set as 'p'.

So, in my case, in planck_fit_sot, I changed the function call from

planck_fit_sot, wave, temp

to

planck_fit_sot, x, p

... and then within the code I changed all the 'wave' to 'x' and 'temp' to 'p[0]'.

This seems to have solved my problem.

best,
Graham

On Thursday, April 2, 2015 at 10:16:52 AM UTC+1, graham kerr wrote:
> Hi,
>
> Yes, that was a typo (but wasn't in my actual code).
>
> start_temp is a float (or double) not a 1-element array so I don't think that's where the error is unfortunately.
>
>
>
>
>
> On Wednesday, April 1, 2015 at 7:14:08 PM UTC+1, Jeremy Bailin wrote:
>> On Tuesday, March 31, 2015 at 5:45:03 PM UTC-5, graham kerr wrote:
>>> Hello everyone,
>>>
>>> I am trying to use mpfitfun to fit data observed at multiple wavelengths to a blackbody function, with temperature as the only variable; so I'm trying to find the best fit temperature.
>>>
>>> My function is called planck_fit_sot.pro, and is below. When I use mpfitfun the output has clearly only tried to fit the first data point. For a few test runs where I simulated blackbody intensities at multiple wavelengths (100 in total), the fitting routine returns the temperature that I set the first data point to. Also, yfit has only one value (the first), with all the rest '0'.
>>>

>>> Does anyone know what (presumably silly) mistake I've made here, and why mpfitfun is not using all the data to fit the function?
>>>
>>> cheers,
>>> Graham
>>> ________________________________
>>>
>>> mpfitfun procedure where wave_rgb & data_rgb are input and temp_range and start_temp are included as optional input :-
>>>

```
>>> if n_elements(start_temp) eq 0 then start_temp = double(6000.0)
>>> parinfo = {value:0.0, fixed:0, limited:[0,0], limits:[0.0,0.0]}
>>>   parinfo[0].value = start_temp
>>>   parinfo[0].fixed = 0
>>>   if n_elements(temp_range) eq 0 then begin
>>>       parinfo[0].limited(*) = 0
>>>   endif else begin
>>>       parinfo[0].limited(*) = 1
>>>       parinfo[0].limits[0] = temp_range[0]
>>>       parinfo[0].limits[1] = temp_range[1]
>>>   endelse
>>>
>>> fit_fn = mpfitfun('planck_fit_sot', wave_rgb, data_rgb, err, $
>>>                  parinfo = parinfo, double = double,$
>>>                  maxiter = 2000, bestnorm = bestnorm,$
>>>                  yfit = yfit, perror = perror, dof = dof,$
>>>                  status = status, errmsg=errmsg)
>>> ________________________________
>>>
>>> planck_fit_sot.pro :-
>>>
>>> FUNCTION planck_fit_sot, wave, temp
>>>
>>>    ;Some constants
>>>    cc = 2.99792458d10    ;cm/s
>>>    hh = 6.62606957d-27  ;erg s
>>>    kb = 1.3806488d-16   ;erg/K
>>>
>>>    wave_cm = wave/1.e8  ;cm
>>>
>>>    bb_fn = dblarr(n_elements(wave))
>>>
>>>  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ;;;;;;;
>>> ;;;;;;;;;;     DEFINE THE FUNCTION    ;;;;;;;;
>>>  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;; ;;;;;;;
>>>
>>>   ;2*h*c^2.0
>>>   const1 = double(2*hh*cc*cc)
```

>>>
>>>   ;h*c/k
>>>   const2 = double(hh*cc/kb)/wave_cm
>>>
>>>   ;; ;;;;;;;
>>>
>>>   bb_fn = const1 / (wave_cm^5.0 * (exp(const2]/temp)-1.))
>>>
>>>   bb_fn = bb_fn*1.d-8 ;ergs/s/cm^2/sr/Ang
>>>
>>>   bb_fn_watts = bb_fn/1.e7 ;W/cm^2/sr/Ang
>>>   ;; ;;;;;;;
>>>
>>>   return, bb_fn_watts
>>>
>>> end
>>
>>
>> As written it won't compile -- I'm guessing the "]" isn't supposed to be here:
>>
>>   bb_fn = const1 / (wave_cm^5.0 * (exp(const2]/temp)-1.))
>>
>> Assuming that's fixed, I would speculate that your start_temp variable coming into the function is an array (possibly a 1-element array) instead of a scalar. Try "help, start_temp" to check.
>>
>> -Jeremy.