Subject: Uneven plot symbol sizes
Posted by on Thu, 16 Apr 2015 11:24:46 GMT
View Forum Message <> Reply to Message

The most common plot symbol sizes seem to be normalized to a common width, which makes e.g. x symbols come out looking sqrt(2) times larger than + symbols. Same thing with squares vs. diamonds - with circles somewhere in between.

I was sufficiently annoyed by this that I wrote a function that measures "areas" of the different symbols defined by cgsymcat. The function returns the real area of the filled symbols normalized to circles (measured by plotting a single, large, white symbol on a black background with no axes, and measuring the total of the resulting array). For the non-filled symbols the area of the corresponding filled symbols, like filled square for open square and x, filled diamond for open diamond and +, etc.

If I then plot with the nominal symsize I want divided by the square root of the area for the particular symbols I'm using, the plots look alright to me.

Now that I got it working, I'm wondering if I've just reinvented an existing wheel. Is this something people worry about enough to have fixed it, preferably in a more elegant way than I did it...

Subject: Re: Uneven plot symbol sizes
Posted by David Fanning on Thu, 16 Apr 2015 13:07:56 GMT
View Forum Message <> Reply to Message

Mats Löfdahl writes:

>
> The most common plot symbol sizes seem to be normalized to a common width, which makes e.g. x symbols come out looking sqrt(2) times larger than + symbols. Same thing with squares vs. diamonds - with circles somewhere in between.
>
> I was sufficiently annoyed by this that I wrote a function that measures "areas" of the different symbols defined by cgsymcat. The function returns the real area of the filled symbols normalized to circles (measured by plotting a single, large, white symbol on a black background with no axes, and measuring the total of the resulting array). For the non-filled symbols the area of the corresponding filled symbols, like filled square for open square and x, filled diamond for open diamond and +, etc.
>
> If I then plot with the nominal symsize I want divided by the square root of the area for the particular symbols I'm using, the plots look alright to me.
>
> Now that I got it working, I'm wondering if I've just reinvented an existing wheel. Is this something people worry about enough to have fixed it, preferably in a more elegant way than I did it...

I have to admit this is one graphical problem that has somehow avoided

my worry list. But, it also sounds like a good idea. If it is easy to
implement, I would consider adding it (probably as a keyword switch) to
cgSymCat.

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.")

Subject: Re: Uneven plot symbol sizes
Posted by on Thu, 16 Apr 2015 13:54:10 GMT
View Forum Message <> Reply to Message

Den torsdag 16 april 2015 kl. 15:08:01 UTC+2 skrev David Fanning:
> Mats Löfdahl writes:
>
>>
>> The most common plot symbol sizes seem to be normalized to a common width, which makes e.g. x symbols come out looking sqrt(2) times larger than + symbols. Same thing with squares vs. diamonds - with circles somewhere in between.
>>
>> I was sufficiently annoyed by this that I wrote a function that measures "areas" of the different symbols defined by cgsymcat. The function returns the real area of the filled symbols normalized to circles (measured by plotting a single, large, white symbol on a black background with no axes, and measuring the total of the resulting array). For the non-filled symbols the area of the corresponding filled symbols, like filled square for open square and x, filled diamond
> for open diamond and +, etc.
>>
>> If I then plot with the nominal symsize I want divided by the square root of the area for the particular symbols I'm using, the plots look alright to me.
>>
>> Now that I got it working, I'm wondering if I've just reinvented an existing wheel. Is this something people worry about enough to have fixed it, preferably in a more elegant way than I did it...
>
> I have to admit this is one graphical problem that has somehow avoided
> my worry list. But, it also sounds like a good idea. If it is easy to
> implement, I would consider adding it (probably as a keyword switch) to
> cgSymCat.

David, that would be fantastic! Then it would come out right in cglegend as well, which I couldn't easily do since it requires symsize to be a scalar.

Implementing it should be easy enough; once you have decided on the scaling, it's just one number per symbol. My numbers are below, you may feel differently about some of them.

Basically, what I wanted was equal length of the lines in x and +, and equal blackness (or whatever color you are using) weight for the filled symbols. And the open symbols the same size as the filled ones. So those were easy to decide. The "hat up" etc. symbols don't have filled versions so I scaled them as diamonds.

For the half circles I decided to deviate from the equal blackness idea and instead scale them as the full circles. I have never used them myself but I guess if you plotted with half circles and circles in the same figure you'd want them to have the same linear size?

For 0, 3, 8, and 10 I just return unity, this area calculation doesn't really apply to them.

My data (i, 1/sqrt(area[i])):

0	1.00000
1	1.25002
2	0.884060
3	1.00000
4	1.25002
5	1.24985
6	0.884060
7	0.884060
8	1.00000
9	1.00000
10	1.00000
11	1.25013
12	1.24987
13	1.24985
14	1.25002
15	0.884060
16	1.00000
17	1.24985
18	1.25013
19	1.24987
20	1.24985
21	1.24961
22	1.24961
23	1.24947
24	1.24947
25	1.24988
26	1.24988
27	1.24991
28	1.24991
29	1.25002
30	1.25002
31	1.25002

32	1.25002
33	1.23751
34	1.23751
35	1.00000
36	1.00000
37	1.00000
38	1.00000
39	1.00000
40	1.00000
41	1.00000
42	1.00000
43	1.00000
44	1.00000
45	1.53805
46	1.53805

Subject: Re: Uneven plot symbol sizes
Posted by on Thu, 16 Apr 2015 14:19:13 GMT
View Forum Message <> Reply to Message

Den torsdag 16 april 2015 kl. 15:54:11 UTC+2 skrev Mats Löfdahl:
>
> Then it would come out right in cglegend as well...

Well, it would if cglegend gets that same keyword switch. :)
