Subject: Interpolation routines
Posted by laura.hike on Tue, 02 Feb 2016 00:47:15 GMT
View Forum Message <> Reply to Message

I was wondering whether anyone could recommend an IDL routine (or extra keywords) that would allow me to interpolate my data correctly. Here are the constraints:

- input data and output data are on irregular 1D grids
- NaN is used as the indicator or bad or missing values
- the input abcissa values always span a broader range than the output values, HOWEVER, sometimes not all of the input locations have ordinate values associated with them. This generally occurs at the top or bottom of the column. In other words, a number of the levels at the top or bottom may be filled with NaNs.

What I'm looking for is a higher order (not linear) interpolation routine that deals gracefully with NaNs and won't interpolate beyond the range of known values.

I tried interpol with /spline. If I use the /NaN option, the program seems to run fine but fails when there are fewer than 4 good values in a row. If I take out the /NaN, it does not fail in this case. However, it turns out that it interpolates beyond where there are any valid input values (i.e., out into the NaN area). If I replace the NaNs with zeros, it does the same thing (which surprised me).

I tried the spline command instead. It failed when I substituted 0s for the NaNs and returned all NaNs when I left them in.

Rather than continuing to try all of IDL's interpolation routines with all their options, I was hoping someone would be able to answer this question off the top of their head.

Thanks,

Laura

Subject: Re: Interpolation routines
Posted by Craig Markwardt on Tue, 02 Feb 2016 17:34:14 GMT
View Forum Message <> Reply to Message

On Monday, February 1, 2016 at 7:47:21 PM UTC-5, laura...@gmail.com wrote:
> I was wondering whether anyone could recommend an IDL routine (or extra keywords) that would allow me to interpolate my data correctly. Here are the constraints:
>
> - input data and output data are on irregular 1D grids
> - NaN is used as the indicator or bad or missing values
> - the input abcissa values always span a broader range than the output values, HOWEVER, sometimes not all of the input locations have ordinate values associated with them. This generally occurs at the top or bottom of the column. In other words, a number of the levels at the top or bottom may be filled with NaNs.
>

> What I'm looking for is a higher order (not linear) interpolation routine that deals gracefully with NaNs and won't interpolate beyond the range of known values.
>
> I tried interpol with /spline. If I use the /NaN option, the program seems to run fine but fails when there are fewer than 4 good values in a row. If I take out the /NaN, it does not fail in this case. However, it turns out that it interpolates beyond where there are any valid input values (i.e., out into the NaN area). If I replace the NaNs with zeros, it does the same thing (which surprised me).
>
> I tried the spline command instead. It failed when I substituted 0s for the NaNs and returned all NaNs when I left them in.

I found most of the built-in IDL interpolation routines to be kind of a joke. The one I use often is SPL_INIT and SPL_INTERP, which are based on the Numerical Recipes interpolation routines. You will have to screen out your NANs manually with WHERE(), but that should not be a big deal.

The IDL Astronomy library has some other nice routines like QUADTERP. I have developed some specialized interpolators for IDL (example: Chebyshev). If you have known explicit derivatives at each sample point, then my CUBETERP or QINTERP might be of use.

Craig

Subject: Re: Interpolation routines
Posted by laura.hike on Tue, 02 Feb 2016 19:18:33 GMT
View Forum Message <> Reply to Message

Ah, so when my friends say I should be using Matlab or (better!) Python, they're right..... ;-)

It's funny that I didn't run into these problems sooner.

Subject: Re: Interpolation routines
Posted by wlandsman on Tue, 02 Feb 2016 20:17:03 GMT
View Forum Message <> Reply to Message

On Tuesday, February 2, 2016 at 12:34:17 PM UTC-5, Craig Markwardt wrote:

>
> I found most of the built-in IDL interpolation routines to be kind of a joke. The one I use often is SPL_INIT and SPL_INTERP, which are based on the Numerical Recipes interpolation routines. You will have to screen out your NANs manually with WHERE(), but that should not be a big deal.

I don't believe this is true anymore. INTERPOL() used to be biggest problem but since IDL 7.0,

it is a professional level interpolation routine (and certainly more powerful than QUADTERP).

I do agree that you best bet is to probably remove the NAN values (using WHERE and FINITE()). I would use INTERPOL() without the /NAN keyword.

g = where(finite(y))
result = interpol(y[g], x[g], xinterp, /spline)

> The IDL Astronomy library has some other nice routines like QUADTERP. I have developed some specialized interpolators for IDL (example: Chebyshev). If you have known explicit derivatives at each sample point, then my CUBETERP or QINTERP might be of use.
>
> Craig

Subject: Re: Interpolation routines
Posted by laura.hike on Tue, 02 Feb 2016 21:31:11 GMT
View Forum Message <> Reply to Message

On Tuesday, February 2, 2016 at 12:17:06 PM UTC-8, wlandsman wrote:

> I don't believe this is true anymore. INTERPOL() used to be biggest problem but since IDL 7.0, it is a professional level interpolation routine (and certainly more powerful than QUADTERP).
>
> I do agree that you best bet is to probably remove the NAN values (using WHERE and FINITE()). I would use INTERPOL() without the /NAN keyword.
>
> g = where(finite(y))
> result = interpol(y[g], x[g], xinterp, /spline)

Thanks for the comment, wlandsman. I tried the "finite" method and got the same result as when I left the NaNs in and included the /NaN keyword. The interpolation works correctly (except when there are fewer than 4 valid points left, so I would have to write a trap for this in any case), but it continues beyond the end of the input range, becoming extrapolation instead of interpolation. This is completely invalid, so I specifically want to prevent it. I show an example below in order to be clear. Yes, this is a vertical profile in pressure coordinates, for anyone who cares:

Input abscissa Pin:

0.187500 0.362500 0.675000 1.26250 2.02500 2.98750 4.47500 6.75000
8.40000 8.98750 9.62500 10.2875 11.0000 11.7750 12.6000 13.4875
14.4500 15.4750 16.5750 17.7625 19.0250 20.3875 21.8625 23.4375
25.1250 26.9375 28.8875 30.9750 33.2125 35.6125 38.1875 40.9750
43.9750 46.3375 48.0000 49.7250 51.5125 53.3625 55.2875 57.2875
59.3500 61.4875 63.7000 66.0000 68.3875 70.8625 73.4250 76.0750
78.8250 81.6875 84.6500 87.7125 90.9000 94.2000 97.6125 101.150

104.825 108.637 112.587 116.688 120.938 125.337 129.900 134.638
139.538 144.625 149.913 155.388 161.062 166.938 173.025 179.350
185.913 192.700 199.737 207.038 214.663 222.625 230.888 239.513
248.513 257.913 267.725 277.925 288.575 299.688 311.237 323.100
335.288 347.938 361.050 374.463 388.188 402.400 417.025 432.087
447.612 463.550 479.925 496.812 514.300 532.287 550.725 569.738
589.312 609.463 630.287 651.825 NaN NaN NaN NaN NaN
NaN NaN NaN NaN NaN NaN NaN NaN NaN
NaN NaN NaN NaN NaN NaN NaN NaN NaN
NaN NaN NaN NaN NaN NaN NaN

Output abscissa Pout:

0.100000 0.300000 0.400000 0.500000 0.700000 1.00000 2.00000
3.00000 4.00000 5.00000 7.00000 10.0000 20.0000 30.0000 40.0000
50.0000 70.0000 100.000 150.000 200.000 250.000 300.000 350.000
400.000 450.000 500.000 550.000 600.000 650.000 700.000 725.000
750.000 775.000 800.000 825.000 850.000 875.000 900.000 925.000
950.000 975.000 1000.00

Values obviously occur in Pout that are beyond the range of Pin.

Ordinate in SWin:

0.000156151 0.000173488 0.000207377 0.000210915 0.000169155 0.000110047
7.45450e-05 5.23441e-05 3.99181e-05 3.90353e-05
3.75340e-05 3.97617e-05 3.25255e-05 3.04963e-05 3.44416e-05 2.90153e-05 2.68362e-05
3.02042e-05 2.54594e-05 2.58893e-05
2.62765e-05 2.39674e-05 2.39971e-05 2.40214e-05 2.37010e-05 2.34202e-05 2.16869e-05
2.26950e-05 1.99376e-05 2.16858e-05
1.91526e-05 1.85088e-05 1.88882e-05 1.80156e-05 1.72226e-05 1.67287e-05 1.73794e-05
1.56135e-05 1.48229e-05 1.44584e-05
1.51021e-05 1.45813e-05 1.19285e-05 1.34967e-05 1.10665e-05 1.15942e-05 1.21990e-05
1.08427e-05 9.58437e-06 1.00086e-05
8.94639e-06 9.36905e-06 8.25729e-06 8.73890e-06 7.72265e-06 7.45533e-06 6.50589e-06
6.92547e-06 6.66663e-06 6.42854e-06
6.20490e-06 5.99692e-06 5.77185e-06 5.05578e-06 4.90454e-06 4.69174e-06 4.99278e-06
4.37616e-06 3.80173e-06 3.67547e-06
3.93453e-06 3.40434e-06 3.28958e-06 3.18232e-06 3.06034e-06 2.62840e-06 2.80613e-06
2.71086e-06 2.02691e-06 2.48816e-06
1.59571e-06 2.53445e-06 2.43971e-06 3.04975e-06 4.47653e-06 4.95492e-06 6.00842e-06
7.14583e-06 7.06873e-06 7.76889e-06
7.85854e-06 8.32823e-06 8.90338e-06 9.45487e-06 9.72611e-06 9.55187e-06 9.46401e-06
8.91291e-06 9.40596e-06 8.94922e-06
9.59435e-06 9.93338e-06 9.91535e-06 1.02259e-05 1.02142e-05 1.00091e-05 8.75643e-06
8.13227e-06 NaN NaN
NaN NaN NaN NaN NaN NaN NaN NaN NaN
NaN

NaN NaN NaN NaN NaN NaN NaN NaN NaN
NaN
NaN NaN NaN NaN NaN NaN NaN NaN

Ordinate out:

0.000147821 0.000166955 0.000177699 0.000189441 0.000209155 0.000217363
0.000170853 0.000109465 8.07843e-05 6.65560e-05
4.99181e-05 3.92060e-05 2.45898e-05 2.22732e-05 1.85323e-05 1.68197e-05 1.12288e-05
7.54906e-06 4.98966e-06 3.04681e-06
1.66306e-06 4.97638e-06 7.80974e-06 9.37925e-06 9.38277e-06 9.00669e-06 9.91276e-06
1.01958e-05 8.16361e-06 4.44220e-06
-3.16524e-06 -1.78092e-05 -4.18935e-05 -7.78224e-05 -0.000128000 -0.000194829
-0.000280715 -0.000388062 -0.000519273 -0.000676752
-0.000862904 -0.00108013

Values are returned for every Pout, whether it's off the end of Pin or not. If you plot these, they form an arc in the negative going to magnitudes higher than any of the input values. Is there any way to get rid of these short of searching for the max and min values of Pin and excluding Pouts outside this range? How does interpol come up with values out there anyway, given that inputs on either side of the output should be required for the spline function? Adding all these searches and exclusions would make using this function impractical, given that I have to apply it thousands of times. Note that Pin varies over time while Pout is fixed.

Subject: Re: Interpolation routines
Posted by laura.hike on Tue, 02 Feb 2016 21:48:15 GMT
View Forum Message <> Reply to Message

I also tried substituting zeros for the SWins where they and Pin are NaNs or only substituting zeros for the NaNs in Pin, and got the same result as with the NaNs in both cases. If I substitute zeros for the NaNs in both SWin and Pin, the result is nonsensical.

Subject: Re: Interpolation routines
Posted by Paul Van Delst[1] on Tue, 02 Feb 2016 22:01:29 GMT
View Forum Message <> Reply to Message

On 02/02/16 16:31, laura.hike@gmail.com wrote:
> On Tuesday, February 2, 2016 at 12:17:06 PM UTC-8, wlandsman wrote:
>
>> I don't believe this is true anymore. INTERPOL() used to be
>> biggest problem but since IDL 7.0, it is a professional level
>> interpolation routine (and certainly more powerful than QUADTERP).
>>
>> I do agree that you best bet is to probably remove the NAN values
>> (using WHERE and FINITE()). I would use INTERPOL() without the

>> /NAN keyword.
>>
>> g = where(finite(y)) result = interpol(y[g], x[g], xinterp,
>> /spline)
>
> Thanks for the comment, wlandsman. I tried the "finite" method and
> got the same result as when I left the NaNs in and included the /NaN
> keyword. The interpolation works correctly (except when there are
> fewer than 4 valid points left, so I would have to write a trap for
> this in any case), but it continues beyond the end of the input
> range, becoming extrapolation instead of interpolation. This is
> completely invalid, so I specifically want to prevent it.

Umm, don't you specify the output interpolated pressures? So shouldn't you just truncate them based on the inputs? That is, the maximum output/interpolated pressure is the first value that is less than the maximum finite input pressure. (Does that sound right?)

I interpolate lots of atmospheric temperature and absorber (e.g. H2O, O3, etc) profiles and that's what I do. I agree with you that extrapolation is completely invalid (in general), and in my case for T(p) and H2O(p)/etc profiles it gets non-physical real fast.

Also, I don't know what the "SW" data is but, for atmospheric profiles, shouldn't you be interpolating in ln(P) space rather than P?

> I show an
> example below in order to be clear. Yes, this is a vertical profile
> in pressure coordinates, for anyone who cares:
>
> Input abscissa Pin:
>
> 0.187500 0.362500 0.675000 1.26250 2.02500
> 2.98750 4.47500 6.75000 8.40000 8.98750 9.62500
> 10.2875 11.0000 11.7750 12.6000 13.4875
> 14.4500 15.4750 16.5750 17.7625 19.0250
> 20.3875 21.8625 23.4375 25.1250 26.9375
> 28.8875 30.9750 33.2125 35.6125 38.1875
> 40.9750 43.9750 46.3375 48.0000 49.7250
> 51.5125 53.3625 55.2875 57.2875 59.3500
> 61.4875 63.7000 66.0000 68.3875 70.8625
> 73.4250 76.0750 78.8250 81.6875 84.6500
> 87.7125 90.9000 94.2000 97.6125 101.150
> 104.825 108.637 112.587 116.688 120.938
> 125.337 129.900 134.638 139.538 144.625
> 149.913 155.388 161.062 166.938 173.025
> 179.350 185.913 192.700 199.737 207.038

> 214.663 222.625 230.888 239.513 248.513 257.913
> 267.725 277.925 288.575 299.688 311.237
> 323.100 335.288 347.938 361.050 374.463
> 388.188 402.400 417.025 432.087 447.612
> 463.550 479.925 496.812 514.300 532.287
> 550.725 569.738 589.312 609.463 630.287
> 651.825 NaN NaN NaN NaN NaN
> NaN NaN NaN NaN NaN NaN
> NaN NaN NaN NaN NaN NaN
> NaN NaN NaN NaN NaN NaN
> NaN NaN NaN NaN NaN NaN
> NaN
>
> Output abscissa Pout:
>
> 0.100000 0.300000 0.400000 0.500000 0.700000
> 1.00000 2.00000 3.00000 4.00000 5.00000 7.00000
> 10.0000 20.0000 30.0000 40.0000 50.0000
> 70.0000 100.000 150.000 200.000 250.000
> 300.000 350.000 400.000 450.000 500.000
> 550.000 600.000 650.000 700.000 725.000 750.000
> 775.000 800.000 825.000 850.000 875.000
> 900.000 925.000 950.000 975.000 1000.00
>
> Values obviously occur in Pout that are beyond the range of Pin.
>
> Ordinate in SWin:
>
> 0.000156151 0.000173488 0.000207377 0.000210915 0.000169155
> 0.000110047 7.45450e-05 5.23441e-05 3.99181e-05 3.90353e-05
> 3.75340e-05 3.97617e-05 3.25255e-05 3.04963e-05 3.44416e-05
> 2.90153e-05 2.68362e-05 3.02042e-05 2.54594e-05 2.58893e-05
> 2.62765e-05 2.39674e-05 2.39971e-05 2.40214e-05 2.37010e-05
> 2.34202e-05 2.16869e-05 2.26950e-05 1.99376e-05 2.16858e-05
> 1.91526e-05 1.85088e-05 1.88882e-05 1.80156e-05 1.72226e-05
> 1.67287e-05 1.73794e-05 1.56135e-05 1.48229e-05 1.44584e-05
> 1.51021e-05 1.45813e-05 1.19285e-05 1.34967e-05 1.10665e-05
> 1.15942e-05 1.21990e-05 1.08427e-05 9.58437e-06 1.00086e-05
> 8.94639e-06 9.36905e-06 8.25729e-06 8.73890e-06 7.72265e-06
> 7.45533e-06 6.50589e-06 6.92547e-06 6.66663e-06 6.42854e-06
> 6.20490e-06 5.99692e-06 5.77185e-06 5.05578e-06 4.90454e-06
> 4.69174e-06 4.99278e-06 4.37616e-06 3.80173e-06 3.67547e-06
> 3.93453e-06 3.40434e-06 3.28958e-06 3.18232e-06 3.06034e-06
> 2.62840e-06 2.80613e-06 2.71086e-06 2.02691e-06 2.48816e-06
> 1.59571e-06 2.53445e-06 2.43971e-06 3.04975e-06 4.47653e-06
> 4.95492e-06 6.00842e-06 7.14583e-06 7.06873e-06 7.76889e-06
> 7.85854e-06 8.32823e-06 8.90338e-06 9.45487e-06 9.72611e-06
> 9.55187e-06 9.46401e-06 8.91291e-06 9.40596e-06 8.94922e-06

> 9.59435e-06 9.93338e-06 9.91535e-06 1.02259e-05 1.02142e-05
> 1.00091e-05 8.75643e-06 8.13227e-06 NaN NaN NaN
> NaN NaN NaN NaN NaN NaN
> NaN NaN NaN NaN NaN NaN
> NaN NaN NaN NaN NaN NaN
> NaN NaN NaN NaN NaN NaN
> NaN NaN NaN
>
> Ordinate out:
>
> 0.000147821 0.000166955 0.000177699 0.000189441 0.000209155
> 0.000217363 0.000170853 0.000109465 8.07843e-05 6.65560e-05
> 4.99181e-05 3.92060e-05 2.45898e-05 2.22732e-05 1.85323e-05
> 1.68197e-05 1.12288e-05 7.54906e-06 4.98966e-06 3.04681e-06
> 1.66306e-06 4.97638e-06 7.80974e-06 9.37925e-06 9.38277e-06
> 9.00669e-06 9.91276e-06 1.01958e-05 8.16361e-06 4.44220e-06
> -3.16524e-06 -1.78092e-05 -4.18935e-05 -7.78224e-05 -0.000128000
> -0.000194829 -0.000280715 -0.000388062 -0.000519273 -0.000676752
> -0.000862904 -0.00108013
>
> Values are returned for every Pout, whether it's off the end of Pin
> or not. If you plot these, they form an arc in the negative going to
> magnitudes higher than any of the input values. Is there any way to
> get rid of these short of searching for the max and min values of Pin
> and excluding Pouts outside this range? How does interpol come up
> with values out there anyway, given that inputs on either side of the
> output should be required for the spline function? Adding all these
> searches and exclusions would make using this function impractical,
> given that I have to apply it thousands of times. Note that Pin
> varies over time while Pout is fixed.
>
>

Subject: Re: Interpolation routines
Posted by laura.hike on Tue, 02 Feb 2016 22:38:15 GMT
View Forum Message <> Reply to Message

On Tuesday, February 2, 2016 at 2:01:35 PM UTC-8, Paul van Delst wrote:

Yes, this is the only solution that I've found. It seems ridiculous to have to do this manually -- surely the interpolation function should handle this, or at least have it as an option. Cutting the range does have the drawback of not assigning a value to a point that is barely outside the input range. Could throw in a little margin for that, I suppose.

SWin is related to density, which is linear in P.

>
> Umm, don't you specify the output interpolated pressures? So shouldn't
> you just truncate them based on the inputs? That is, the maximum
> output/interpolated pressure is the first value that is less than the
> maximum finite input pressure. (Does that sound right?)
>
> I interpolate lots of atmospheric temperature and absorber (e.g. H2O,
> O3, etc) profiles and that's what I do. I agree with you that
> extrapolation is completely invalid (in general), and in my case for
> T(p) and H2O(p)/etc profiles it gets non-physical real fast.
>
> Also, I don't know what the "SW" data is but, for atmospheric profiles,
> shouldn't you be interpolating in ln(P) space rather than P?
>
>
>> I show an
>> example below in order to be clear. Yes, this is a vertical profile
>> in pressure coordinates, for anyone who cares:
>>
>> Input abscissa Pin:
>>
>> 0.187500 0.362500 0.675000 1.26250 2.02500
>> 2.98750 4.47500 6.75000 8.40000 8.98750 9.62500
>> 10.2875 11.0000 11.7750 12.6000 13.4875
>> 14.4500 15.4750 16.5750 17.7625 19.0250
>> 20.3875 21.8625 23.4375 25.1250 26.9375
>> 28.8875 30.9750 33.2125 35.6125 38.1875
>> 40.9750 43.9750 46.3375 48.0000 49.7250
>> 51.5125 53.3625 55.2875 57.2875 59.3500
>> 61.4875 63.7000 66.0000 68.3875 70.8625
>> 73.4250 76.0750 78.8250 81.6875 84.6500
>> 87.7125 90.9000 94.2000 97.6125 101.150
>> 104.825 108.637 112.587 116.688 120.938
>> 125.337 129.900 134.638 139.538 144.625
>> 149.913 155.388 161.062 166.938 173.025
>> 179.350 185.913 192.700 199.737 207.038
>> 214.663 222.625 230.888 239.513 248.513 257.913
>> 267.725 277.925 288.575 299.688 311.237
>> 323.100 335.288 347.938 361.050 374.463
>> 388.188 402.400 417.025 432.087 447.612
>> 463.550 479.925 496.812 514.300 532.287
>> 550.725 569.738 589.312 609.463 630.287
>> 651.825 NaN NaN NaN NaN NaN
>> NaN NaN NaN NaN NaN NaN
>> NaN NaN NaN NaN NaN NaN
>> NaN NaN NaN NaN NaN NaN
>> NaN NaN NaN NaN NaN NaN

>> NaN
>>
>> Output abscissa Pout:
>>
>> 0.100000 0.300000 0.400000 0.500000 0.700000
>> 1.00000 2.00000 3.00000 4.00000 5.00000 7.00000
>> 10.0000 20.0000 30.0000 40.0000 50.0000
>> 70.0000 100.000 150.000 200.000 250.000
>> 300.000 350.000 400.000 450.000 500.000
>> 550.000 600.000 650.000 700.000 725.000 750.000
>> 775.000 800.000 825.000 850.000 875.000
>> 900.000 925.000 950.000 975.000 1000.00
>>
>> Values obviously occur in Pout that are beyond the range of Pin.
>>
>> Ordinate in SWin:
>>
>> 0.000156151 0.000173488 0.000207377 0.000210915 0.000169155
>> 0.000110047 7.45450e-05 5.23441e-05 3.99181e-05 3.90353e-05
>> 3.75340e-05 3.97617e-05 3.25255e-05 3.04963e-05 3.44416e-05
>> 2.90153e-05 2.68362e-05 3.02042e-05 2.54594e-05 2.58893e-05
>> 2.62765e-05 2.39674e-05 2.39971e-05 2.40214e-05 2.37010e-05
>> 2.34202e-05 2.16869e-05 2.26950e-05 1.99376e-05 2.16858e-05
>> 1.91526e-05 1.85088e-05 1.88882e-05 1.80156e-05 1.72226e-05
>> 1.67287e-05 1.73794e-05 1.56135e-05 1.48229e-05 1.44584e-05
>> 1.51021e-05 1.45813e-05 1.19285e-05 1.34967e-05 1.10665e-05
>> 1.15942e-05 1.21990e-05 1.08427e-05 9.58437e-06 1.00086e-05
>> 8.94639e-06 9.36905e-06 8.25729e-06 8.73890e-06 7.72265e-06
>> 7.45533e-06 6.50589e-06 6.92547e-06 6.66663e-06 6.42854e-06
>> 6.20490e-06 5.99692e-06 5.77185e-06 5.05578e-06 4.90454e-06
>> 4.69174e-06 4.99278e-06 4.37616e-06 3.80173e-06 3.67547e-06
>> 3.93453e-06 3.40434e-06 3.28958e-06 3.18232e-06 3.06034e-06
>> 2.62840e-06 2.80613e-06 2.71086e-06 2.02691e-06 2.48816e-06
>> 1.59571e-06 2.53445e-06 2.43971e-06 3.04975e-06 4.47653e-06
>> 4.95492e-06 6.00842e-06 7.14583e-06 7.06873e-06 7.76889e-06
>> 7.85854e-06 8.32823e-06 8.90338e-06 9.45487e-06 9.72611e-06
>> 9.55187e-06 9.46401e-06 8.91291e-06 9.40596e-06 8.94922e-06
>> 9.59435e-06 9.93338e-06 9.91535e-06 1.02259e-05 1.02142e-05
>> 1.00091e-05 8.75643e-06 8.13227e-06 NaN NaN NaN
>> NaN NaN NaN NaN NaN NaN
>> NaN NaN NaN NaN NaN NaN
>> NaN NaN NaN NaN NaN NaN
>> NaN NaN NaN NaN NaN NaN
>> NaN NaN NaN
>>
>> Ordinate out:
>>
>> 0.000147821 0.000166955 0.000177699 0.000189441 0.000209155

>> 0.000217363 0.000170853 0.000109465 8.07843e-05 6.65560e-05
>> 4.99181e-05 3.92060e-05 2.45898e-05 2.22732e-05 1.85323e-05
>> 1.68197e-05 1.12288e-05 7.54906e-06 4.98966e-06 3.04681e-06
>> 1.66306e-06 4.97638e-06 7.80974e-06 9.37925e-06 9.38277e-06
>> 9.00669e-06 9.91276e-06 1.01958e-05 8.16361e-06 4.44220e-06
>> -3.16524e-06 -1.78092e-05 -4.18935e-05 -7.78224e-05 -0.000128000
>> -0.000194829 -0.000280715 -0.000388062 -0.000519273 -0.000676752
>> -0.000862904 -0.00108013
>>
>> Values are returned for every Pout, whether it's off the end of Pin
>> or not. If you plot these, they form an arc in the negative going to
>> magnitudes higher than any of the input values. Is there any way to
>> get rid of these short of searching for the max and min values of Pin
>> and excluding Pouts outside this range? How does interpol come up
>> with values out there anyway, given that inputs on either side of the
>> output should be required for the spline function? Adding all these
>> searches and exclusions would make using this function impractical,
>> given that I have to apply it thousands of times. Note that Pin
>> varies over time while Pout is fixed.
>>
>>

Subject: Re: Interpolation routines
Posted by lecacheux.alain on Wed, 03 Feb 2016 09:34:05 GMT
View Forum Message <> Reply to Message

Le mardi 2 février 2016 23:38:18 UTC+1, laura...@gmail.com a écrit :
> On Tuesday, February 2, 2016 at 2:01:35 PM UTC-8, Paul van Delst wrote:
>
> Yes, this is the only solution that I've found. It seems ridiculous to have to do this manually -- surely the interpolation function should handle this, or at least have it as an option. Cutting the range does have the drawback of not assigning a value to a point that is barely outside the input range. Could throw in a little margin for that, I suppose.
>
> SWin is related to density, which is linear in P.
>
>
>

Spline is well known for extrapolating badly (even extremely badly !).
In your case, assuming some stationarity of your data, an autoregressive model (e.g. TS_FCAST in IDL) might give better result (of course after truncating your series by using some where(finite(...)).
alx.
>>
>> Umm, don't you specify the output interpolated pressures? So shouldn't
>> you just truncate them based on the inputs? That is, the maximum

>> output/interpolated pressure is the first value that is less than the
>> maximum finite input pressure. (Does that sound right?)
>>
>> I interpolate lots of atmospheric temperature and absorber (e.g. H2O,
>> O3, etc) profiles and that's what I do. I agree with you that
>> extrapolation is completely invalid (in general), and in my case for
>> T(p) and H2O(p)/etc profiles it gets non-physical real fast.
>>
>> Also, I don't know what the "SW" data is but, for atmospheric profiles,
>> shouldn't you be interpolating in ln(P) space rather than P?
>>
>>
>>> I show an
>>> example below in order to be clear. Yes, this is a vertical profile
>>> in pressure coordinates, for anyone who cares:
>>>
>>> Input abscissa Pin:
>>>
>>> 0.187500 0.362500 0.675000 1.26250 2.02500
>>> 2.98750 4.47500 6.75000 8.40000 8.98750 9.62500
>>> 10.2875 11.0000 11.7750 12.6000 13.4875
>>> 14.4500 15.4750 16.5750 17.7625 19.0250
>>> 20.3875 21.8625 23.4375 25.1250 26.9375
>>> 28.8875 30.9750 33.2125 35.6125 38.1875
>>> 40.9750 43.9750 46.3375 48.0000 49.7250
>>> 51.5125 53.3625 55.2875 57.2875 59.3500
>>> 61.4875 63.7000 66.0000 68.3875 70.8625
>>> 73.4250 76.0750 78.8250 81.6875 84.6500
>>> 87.7125 90.9000 94.2000 97.6125 101.150
>>> 104.825 108.637 112.587 116.688 120.938
>>> 125.337 129.900 134.638 139.538 144.625
>>> 149.913 155.388 161.062 166.938 173.025
>>> 179.350 185.913 192.700 199.737 207.038
>>> 214.663 222.625 230.888 239.513 248.513 257.913
>>> 267.725 277.925 288.575 299.688 311.237
>>> 323.100 335.288 347.938 361.050 374.463
>>> 388.188 402.400 417.025 432.087 447.612
>>> 463.550 479.925 496.812 514.300 532.287
>>> 550.725 569.738 589.312 609.463 630.287
>>> 651.825 NaN NaN NaN NaN NaN
>>> NaN NaN NaN NaN NaN NaN
>>> NaN NaN NaN NaN NaN NaN
>>> NaN NaN NaN NaN NaN NaN
>>> NaN NaN NaN NaN NaN NaN
>>> NaN
>>>
>>> Output abscissa Pout:
>>>

>>> 0.100000 0.300000 0.400000 0.500000 0.700000
>>> 1.00000 2.00000 3.00000 4.00000 5.00000 7.00000
>>> 10.0000 20.0000 30.0000 40.0000 50.0000
>>> 70.0000 100.000 150.000 200.000 250.000
>>> 300.000 350.000 400.000 450.000 500.000
>>> 550.000 600.000 650.000 700.000 725.000 750.000
>>> 775.000 800.000 825.000 850.000 875.000
>>> 900.000 925.000 950.000 975.000 1000.00
>>>
>>> Values obviously occur in Pout that are beyond the range of Pin.
>>>
>>> Ordinate in SWin:
>>>
>>> 0.000156151 0.000173488 0.000207377 0.000210915 0.000169155
>>> 0.000110047 7.45450e-05 5.23441e-05 3.99181e-05 3.90353e-05
>>> 3.75340e-05 3.97617e-05 3.25255e-05 3.04963e-05 3.44416e-05
>>> 2.90153e-05 2.68362e-05 3.02042e-05 2.54594e-05 2.58893e-05
>>> 2.62765e-05 2.39674e-05 2.39971e-05 2.40214e-05 2.37010e-05
>>> 2.34202e-05 2.16869e-05 2.26950e-05 1.99376e-05 2.16858e-05
>>> 1.91526e-05 1.85088e-05 1.88882e-05 1.80156e-05 1.72226e-05
>>> 1.67287e-05 1.73794e-05 1.56135e-05 1.48229e-05 1.44584e-05
>>> 1.51021e-05 1.45813e-05 1.19285e-05 1.34967e-05 1.10665e-05
>>> 1.15942e-05 1.21990e-05 1.08427e-05 9.58437e-06 1.00086e-05
>>> 8.94639e-06 9.36905e-06 8.25729e-06 8.73890e-06 7.72265e-06
>>> 7.45533e-06 6.50589e-06 6.92547e-06 6.66663e-06 6.42854e-06
>>> 6.20490e-06 5.99692e-06 5.77185e-06 5.05578e-06 4.90454e-06
>>> 4.69174e-06 4.99278e-06 4.37616e-06 3.80173e-06 3.67547e-06
>>> 3.93453e-06 3.40434e-06 3.28958e-06 3.18232e-06 3.06034e-06
>>> 2.62840e-06 2.80613e-06 2.71086e-06 2.02691e-06 2.48816e-06
>>> 1.59571e-06 2.53445e-06 2.43971e-06 3.04975e-06 4.47653e-06
>>> 4.95492e-06 6.00842e-06 7.14583e-06 7.06873e-06 7.76889e-06
>>> 7.85854e-06 8.32823e-06 8.90338e-06 9.45487e-06 9.72611e-06
>>> 9.55187e-06 9.46401e-06 8.91291e-06 9.40596e-06 8.94922e-06
>>> 9.59435e-06 9.93338e-06 9.91535e-06 1.02259e-05 1.02142e-05
>>> 1.00091e-05 8.75643e-06 8.13227e-06 NaN NaN NaN
>>> NaN NaN NaN NaN NaN NaN
>>> NaN NaN NaN NaN NaN NaN
>>> NaN NaN NaN NaN NaN NaN
>>> NaN NaN NaN NaN NaN NaN
>>> NaN NaN NaN
>>>
>>> Ordinate out:
>>>
>>> 0.000147821 0.000166955 0.000177699 0.000189441 0.000209155
>>> 0.000217363 0.000170853 0.000109465 8.07843e-05 6.65560e-05
>>> 4.99181e-05 3.92060e-05 2.45898e-05 2.22732e-05 1.85323e-05
>>> 1.68197e-05 1.12288e-05 7.54906e-06 4.98966e-06 3.04681e-06
>>> 1.66306e-06 4.97638e-06 7.80974e-06 9.37925e-06 9.38277e-06

>>> 9.00669e-06 9.91276e-06 1.01958e-05 8.16361e-06 4.44220e-06
>>> -3.16524e-06 -1.78092e-05 -4.18935e-05 -7.78224e-05 -0.000128000
>>> -0.000194829 -0.000280715 -0.000388062 -0.000519273 -0.000676752
>>> -0.000862904 -0.00108013
>>>
>>> Values are returned for every Pout, whether it's off the end of Pin
>>> or not. If you plot these, they form an arc in the negative going to
>>> magnitudes higher than any of the input values. Is there any way to
>>> get rid of these short of searching for the max and min values of Pin
>>> and excluding Pouts outside this range? How does interpol come up
>>> with values out there anyway, given that inputs on either side of the
>>> output should be required for the spline function? Adding all these
>>> searches and exclusions would make using this function impractical,
>>> given that I have to apply it thousands of times. Note that Pin
>>> varies over time while Pout is fixed.
>>>
>>>