**Neighborhood Width**

neighborhood\_win~ quad\_lowest linear\_lowest median\_lowest quad\_rms linear\_rms median\_rms quad\_r\_sqd linear\_r\_sqd median\_r\_sqd

*<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>*

1 10 0 0 1 0.196 0.196 0.196 0.673 0.673 0.673

2 20 0 0.353 0.647 0.165 0.165 0.165 0.752 0.751 0.752

3 30 0 0.347 0.653 0.143 0.146 0.143 0.830 0.826 0.830

4 40 0 0.313 0.687 0.135 0.143 0.135 0.847 0.832 0.847

5 50 0.0733 0.333 0.593 0.144 0.155 0.144 0.832 0.805 0.831

6 60 0.267 0.273 0.46 0.147 0.167 0.148 0.818 0.771 0.815

Chart, line chart

Description automatically generated

For Landsat sampling interval (SNR = 10, CF = 0.5, SP = 16), best performance at middling window size (~ 40, but similar R2 ~ 0.8 for all > 20)

Competing example which might be useful for justifying our choices…

Chart, line chart

Description automatically generated

This is (SNR=5, CF=0.5, SP=16) with R2

Chart, line chart

Description automatically generated

And here the same, but with RMS instead

neighborhood\_win~ quad\_lowest linear\_lowest median\_lowest quad\_rms linear\_rms median\_rms quad\_r\_sqd linear\_r\_sqd median\_r\_sqd

*<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>*

1 10 0 0 1 0.447 0.447 0.447 0.325 0.325 0.325

2 20 0 0.26 0.74 0.437 0.438 0.437 0.370 0.368 0.370

3 30 0 0.407 0.593 0.356 0.360 0.356 0.485 0.482 0.485

4 40 0 0.5 0.5 0.354 0.354 0.354 0.433 0.448 0.433

5 50 0.0533 0.38 0.567 0.335 0.343 0.335 0.468 0.445 0.467

6 60 0.107 0.367 0.527 0.314 0.326 0.313 0.500 0.479 0.505

Chart, line chart

Description automatically generated

For very noisy datasets (SNR = 2, CF = 0.5, SP = 16) better performance with wide windows – but note even best performance isn’t great (e.g. R2= 0.5)

neighborhood\_win~ quad\_lowest linear\_lowest median\_lowest quad\_rms linear\_rms median\_rms quad\_r\_sqd linear\_r\_sqd median\_r\_sqd

*<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>*

1 10 0.2 0.113 0.687 0.0329 0.0390 0.0318 0.989 0.985 0.990

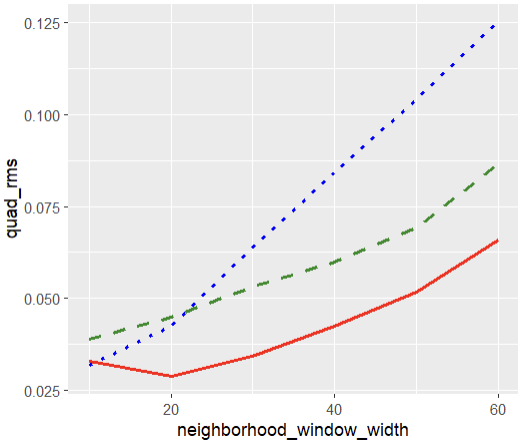
2 20 0.7 0.0733 0.227 0.0288 0.0449 0.0423 0.992 0.981 0.983

3 30 0.827 0.133 0.04 0.0342 0.0530 0.0638 0.988 0.973 0.964

4 40 0.773 0.227 0 0.0423 0.0599 0.0842 0.982 0.965 0.940

5 50 0.767 0.233 0 0.0516 0.0692 0.104 0.973 0.956 0.912

6 60 0.74 0.26 0 0.0660 0.0866 0.125 0.959 0.939 0.874



In better conditions (SNR = 20, CF = 0.5, SP = 2) again quadratic does best, and highest R2 around window of 30 days – although here generally all the outputs were pretty good

**Signal to noise ratio**

signal\_to\_noise\_~ quad\_lowest linear\_lowest median\_lowest quad\_rms linear\_rms median\_rms quad\_r\_sqd linear\_r\_sqd median\_r\_sqd

*<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>*

1 0.5 0.147 0.127 0.727 0.568 0.579 0.501 0.227 0.298 0.341

2 1 0.0133 0.24 0.747 0.334 0.290 0.259 0.463 0.552 0.610

3 2 0.0467 0.24 0.713 0.184 0.161 0.147 0.739 0.795 0.827

4 5 0.413 0.247 0.34 0.0805 0.0862 0.0839 0.942 0.937 0.940

5 10 0.673 0.14 0.187 0.0483 0.0619 0.0674 0.978 0.965 0.959

6 20 0.827 0.133 0.04 0.0342 0.0530 0.0638 0.988 0.973 0.964

Chart

Description automatically generated

This is an interesting test… Here (NW = 30, CF = 0.5, SP = 2) the performance is best with the median method for high-noise situations, but for low noise (SNR >= 5) the quadratic method is best (presumably because it’s not overfitting to noise).

The story isn’t as interesting at other sample intervals or higher cloudiness levels, mainly because there’s too little data and it mostly defaults to the median in all cases.

cloudy\_fraction quad\_lowest linear\_lowest median\_lowest quad\_rms linear\_rms median\_rms quad\_r\_sqd linear\_r\_sqd median\_r\_sqd

*<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>*

1 0 0.393 0.527 0.08 0.0378 0.0342 0.0653 0.986 0.991 0.963

2 0.05 0.447 0.493 0.06 0.0382 0.0364 0.0654 0.986 0.989 0.963

3 0.1 0.58 0.36 0.06 0.0392 0.0398 0.0655 0.985 0.987 0.962

4 0.15 0.547 0.353 0.1 0.0396 0.0409 0.0652 0.985 0.986 0.963

5 0.2 0.547 0.347 0.107 0.0406 0.0431 0.0660 0.984 0.984 0.962

6 0.25 0.56 0.327 0.113 0.0420 0.0459 0.0670 0.983 0.982 0.961

7 0.3 0.653 0.22 0.127 0.0416 0.0491 0.0664 0.984 0.978 0.961

8 0.35 0.693 0.167 0.14 0.0435 0.0517 0.0675 0.982 0.976 0.960

9 0.4 0.68 0.187 0.133 0.0444 0.0518 0.0668 0.981 0.976 0.961

10 0.45 0.7 0.153 0.147 0.0464 0.0561 0.0676 0.980 0.971 0.959

11 0.5 0.673 0.14 0.187 0.0483 0.0619 0.0674 0.978 0.965 0.959

12 0.55 0.633 0.16 0.207 0.0506 0.0663 0.0694 0.977 0.958 0.958

13 0.6 0.66 0.133 0.207 0.0520 0.0686 0.0701 0.975 0.959 0.958

14 0.65 0.64 0.133 0.227 0.0556 0.0728 0.0710 0.971 0.952 0.956

15 0.7 0.62 0.133 0.247 0.0623 0.0809 0.0732 0.965 0.942 0.953

16 0.75 0.567 0.14 0.293 0.0667 0.0856 0.0761 0.958 0.933 0.949

17 0.8 0.46 0.18 0.36 0.0770 0.0936 0.0814 0.946 0.922 0.940

18 0.85 0.427 0.107 0.467 0.0906 0.112 0.0921 0.925 0.891 0.924

19 0.9 0.233 0.26 0.507 0.114 0.126 0.114 0.887 0.865 0.886

20 0.95 0.0133 0.4 0.587 0.178 0.180 0.178 0.741 0.738 0.741

Chart, histogram

Description automatically generated

Here we see that for most cloudiness fractions in a fast-return regime, the quadratic method works best (NW=30, SNR=10, SP=2). Also, the performance here is generally pretty good up until very cloudy scenarios. Even at 95% cloud cover, we had R2 = 0.74. At 85% cloud cover, R2 = 0.93, and it’s higher for less clouds (e.g. R2 = 0.98 for CF=0.5).

cloudy\_fraction quad\_lowest linear\_lowest median\_lowest quad\_rms linear\_rms median\_rms quad\_r\_sqd linear\_r\_sqd median\_r\_sqd

*<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>* *<dbl>*

1 0 0 0.667 0.333 0.0784 0.0711 0.0784 0.948 0.958 0.948

2 0.05 0 0.513 0.487 0.0810 0.0797 0.0810 0.943 0.945 0.943

3 0.1 0 0.447 0.553 0.0833 0.0854 0.0833 0.941 0.937 0.941

4 0.15 0 0.34 0.66 0.0872 0.0919 0.0872 0.933 0.926 0.933

5 0.2 0 0.373 0.627 0.0914 0.0956 0.0914 0.926 0.919 0.926

6 0.25 0 0.327 0.673 0.0944 0.101 0.0944 0.920 0.907 0.920

7 0.3 0 0.32 0.68 0.102 0.107 0.102 0.908 0.901 0.908

8 0.35 0 0.393 0.607 0.109 0.112 0.109 0.899 0.895 0.899

9 0.4 0 0.313 0.687 0.117 0.122 0.117 0.879 0.871 0.879

10 0.45 0 0.287 0.713 0.117 0.123 0.117 0.879 0.870 0.879

11 0.5 0 0.347 0.653 0.143 0.146 0.143 0.830 0.826 0.830

12 0.55 0 0.38 0.62 0.153 0.155 0.153 0.796 0.793 0.796

13 0.6 0 0.34 0.66 0.172 0.175 0.172 0.759 0.754 0.759

14 0.65 0 0.38 0.62 0.182 0.182 0.182 0.730 0.729 0.730

15 0.7 0 0.333 0.667 0.219 0.220 0.219 0.658 0.657 0.658

16 0.75 0 0.193 0.807 0.223 0.223 0.223 0.645 0.644 0.645

17 0.8 0 0.153 0.847 0.259 0.258 0.259 0.574 0.575 0.574

18 0.85 0 0.1 0.9 0.315 0.315 0.315 0.526 0.526 0.526

19 0.9 0 0 1 0.363 0.363 0.363 0.449 0.448 0.449

20 0.95 0 0 1 0.410 0.410 0.410 0.434 0.434 0.434

Chart, line chart

Description automatically generated

In this more challenging, Landsat-like scenario (NW=30, SNR=10, SP=16), all methods are similar. Here, the performance is really good with CF ≤ 30% (R2 ≥ 0.908) and pretty good with CF ≤ 50% (R2 ≥ 0.83), but then declines. We should recommend against using this fitting method with low temporal frequency (e.g. Landsat 16-day interval) if CF > 50%.

Finally, we’ll make plots that gives the tradeoff between cloudy fraction, sample period, and error:

Chart, line chart

Description automatically generated

Chart, line chart

Description automatically generated

These use the settings (NW=30, SNR=10)

As we’d expect, the performance falls off with increasing cloud cover, and more quickly for lower return intervals. The second plot (R2) has horizontal lines indicating thresholds of 0.9 (red) and 0.8 (pink).

Generally, the “Sentinel-2” style in the worst case (5-day return) hits the 0.9 threshold around 70% cloud cover, and the 0.8 threshold a bit above 85% cloud cover.

The “Landsat” style in the worst case (16-day return) hits the 0.9 threshold around 30% cloud cover, and 0.8 around 55% cloud cover.

This is really encouraging!

Chart, line chart

Description automatically generated

Same as above, but for SNR = 5, NW=30

Chart, line chart

Description automatically generated

Same as above, but for SNR = 2, NW = 30

Generally, performance is much worse for very noisy data. But I think this isn’t really a problem for the greenness products. It could be more of a problem for thermal imagery!