1 PLS in TSG

An Irishman, an Englishman and a Scot walked out of a bar, and went home. That's what's in store for you here, The User. We might as well get on with it.

Partial Least Squares, or PLS, is a modelling technique. It's generally used in TSG to model an external scalar measurement (e.g., geochemistry) using spectra. There are two parts to it. First there's a complicated calibration stage (a new screen in TSG) where a model is made, then there's a relatively simple prediction stage (a new kind of TSG scalar) where the model is applied to "unknown" samples.

The model that PLS finds is *linear*. PLS prediction is often done using an iterative algorithm that makes one wonder about this, but it can also take a simple form that reveals the model's linearity: $y=\sum_i [\ \mathbf{m_i} * (\mathbf{x_i} - \mathbf{c_i})\ ]$[1]

In a geological environment, PLS modelling often works indirectly or by association. For example, modelling tiny amounts of gold. This is not a daft idea; it can work by modelling the changes in mineral assemblage that are associated with gold concentration.

1.1 Why?

The most common reason is to have a model that predicts an expensive or inconvenient measurement from spectra, for example, a model that predicts gold % from spectra.

Another reason (often part of the first) is to explore the relationship between spectra and some external measurement. Understanding *why* a model works is always a good thing. For example, a measurement of CO_3. If a decent PLS model can be made then some of the calibration results might help you understand how the model distinguishes carbonates from other minerals that have similar absorptions. This understanding might help you take things a step further and put together a simple model of your own, based on spectral indices. Such a model will be more focussed and probably more robust than the PLS model.

PLS can also be fun to experiment with. If you have the time, try modelling some available scalars (especially imports) with spectra and see if you get a "bite". If you find a good model then it might give you something new to think about.

1.2 This document

Most of it deals with the new PLS calibration screen. There's some advice on putting together a calibration set, a discussion about the spectral subsetting and processing options available, saving & loading sessions, and stuff like that. On page 8 it starts on the cross-validation process, which is central to PLS calibration. Some understanding of what makes PLS tick may be found here. After that, the first four PLS calibration plots are described, along with the data and stats they show. Outlier exclusion is also discussed at this point. On page 22 it gets to the PLS algorithm itself, at last, before moving on to the more geeky plots and sort-of fizzing out.

The PLS prediction scalar is described from page 29 onwards. The prediction algorithms are presented, the PLS session file is revisited, and the mechanics of making and diagnosing a PLS prediction scalar are described.

[1] Where **x** is a spectrum, **c** is the model's offset component, **m** is the model's gain component, and y is the prediction. **x**, **c** and **m** are arrays (number of spectral channels) while y is a scalar. The sum goes over spectral channels.

1.3 Calibration – the 'PLS' screen

Like I said, PLS calibration is generally used in TSG to model an external scalar measurement using spectra. It's a complicated process and an entire TSG screen has been dedicated to it. There are lots of plots and options to help you understand if a model is working and, given that it is, what's driving it.

Here's an example of how things might be done:

- (Preparation.) You need a set of calibration samples. More on this in a moment. You might have them organised nicely in a TSG dataset of their own, or scattered here and there in some big dataset. In the latter case, you should set up a class or mask scalar so that you can get at the cal samples easily.
- Cal samples. Click this button in the PLS screen to select your cal samples from the current dataset. The model will be derived from these samples.
- Inputs. PLS calibration doesn't offer a choice of spectral layers. It takes the first layer (reflectance) and does its own thing. Here's where it does its own thing. (Click this button if you want to do spectral subsetting / resampling, or if you want some spectral processing done.)
- Model this. Select the scalar to model. (This scalar has your "concentrations".)
- Run CV. This will construct a series of models and make lots of plots available.
- Save your PLS session. CV can take some time. If you botch something in your fiddling around (coming up shortly) then you can revert quickly by loading a saved session.
- Fiddle around. Evaluate some plots, select one of the models, and perhaps exclude sample or input outliers. If you exclude any outliers then you'll see that the "Run CV" button goes red. You have to run CV again and evaluate what it gives.
- Save your PLS session again. You can carry on fiddling around with it later or use the saved session for a PLS prediction.

1.3.1 Calibration set

Before going any further, I want to tell you about the set of samples you need for PLS calibration.

You need a set of calibration samples for which there are spectra and "concentrations". (In PLS jargon, a "concentration" is the scalar measurement that you want to model, e.g., a geochemistry result. You need to have this thing imported into a TSG scalar.)

Why look, here's a box of bullets.

- Choose a representative set of calibration samples.
  The calibration set must include all spectrally-active components you might find when dealing with unknowns later on. This is crucial.
- Try your best to get each spectrum and concentration measurement from the same actual sample.
  This is often ~~impo~~ a challenge because a spectrum deals with a patch of rock *surface* (commonly 1cm^2) while a concentration measurement is often derived from a relatively large *volume* of rock. Sometimes it might be worth

measuring several spectra of different parts of the rock and using the average spectrum for calibration.

- Try for an even coverage of the concentration in your calibration samples.
  PLS prediction probably won't extrapolate gracefully. Also, don't come along with a whole stack of samples with, say, mid-point concentrations and just a few samples with other concentrations over the spread. The model will probably be poor with misleadingly high stats (especially R^2).
- Do not make mistakes. Do not select a calibration sample from a foreign environment, do not botch a measurement, and do not pair off a spectrum with a different sample's concentration.
- The calibration samples ought to be like the unknown samples you'll be dealing with later.
  E.g., Don't calibrate on powder spectra if you'll be predicting on rock spectra later, and don't calibrate on nice, fresh-surfaced, hand-picked rocks if you'll be predicting on daggy, "varnished", ordinary rocks. Some such differences may be overcome (to some extent) by careful pre-processing but my advice is that it's best to avoid the problem in the first place.
- For best results, the calibration spectra should be measured in the same way as the unknown spectra.
  e.g., Don't integrate the calibration spectra for 60 seconds and the unknown spectra for 1 second. If the unknown spectra are going to be noisy then the calibration spectra ought to be noisy too, so that the PLS model knows how to extract the good spectral variability from the noise and the PLS predictions' spectral residuals[2] are like the calibration ones.
  Similarly, it is not optimal to measure the calibration spectra with one kind of spectrometer and the unknown spectra with another.
- Don't forget to take care with the concentration measurements, and have realistic expectations given measurement accuracy. PLS will be accepting these as "the truth", unaware that like all measurements, they too have errors. (And let's not forget that they themselves may be interpretations masquerading as measurements.) It will try to model these numbers as they are given. So try to minimise the errors.

1.3.2 Load

Click the Load button to load up a PLS session that was saved earlier.

You will find that you cannot load a PLS session for a "different" dataset; you can only load it for the same dataset that made it.

1.3.3 Save

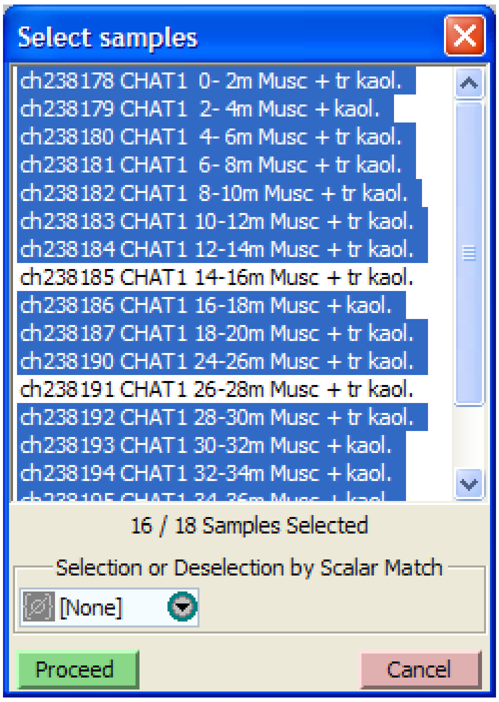
Click the Save button (at almost any time) to save your PLS session. The default filename for the session is made from the dataset's name and the name of the scalar being modelled. This is sensible, I think, but you can use any name you like.

Everything that matters is saved. You can load the session at a later date and pick up where you left off.

A saved session is also the thing that drives a PLS prediction. I'll describe this process later on.

[2] The spectral residual is the only diagnostic available in PLS prediction.

1.3.4 Cal samples



Select samples

ch238178 CHAT1 0- 2m Musc + tr kaol.
ch238179 CHAT1 2- 4m Musc + kaol.
ch238180 CHAT1 4- 6m Musc + tr kaol.
ch238181 CHAT1 6- 8m Musc + tr kaol.
ch238182 CHAT1 8-10m Musc + tr kaol.
ch238183 CHAT1 10-12m Musc + tr kaol.
ch238184 CHAT1 12-14m Musc + tr kaol.
ch238185 CHAT1 14-16m Musc + tr kaol.
ch238186 CHAT1 16-18m Musc + kaol.
ch238187 CHAT1 18-20m Musc + tr kaol.
ch238190 CHAT1 24-26m Musc + tr kaol.
ch238191 CHAT1 26-28m Musc + tr kaol.
ch238192 CHAT1 28-30m Musc + tr kaol.
ch238193 CHAT1 30-32m Musc + kaol.
ch238194 CHAT1 32-34m Musc + kaol.
ch238195 CHAT1 34-36m Musc + kaol.

16 / 18 Samples Selected

Selection or Deselection by Scalar Match: [None]

The PLS screen starts life in ignorance. It does not expect that all samples in the dataset are calibration samples. You have to tell it exactly which samples are, and this is where you do it.

As is usual with multiple-selection lists in TSG, you can use <CTRL>click to toggle an individual selection, or <SHFT>click pairs to select a block of items.

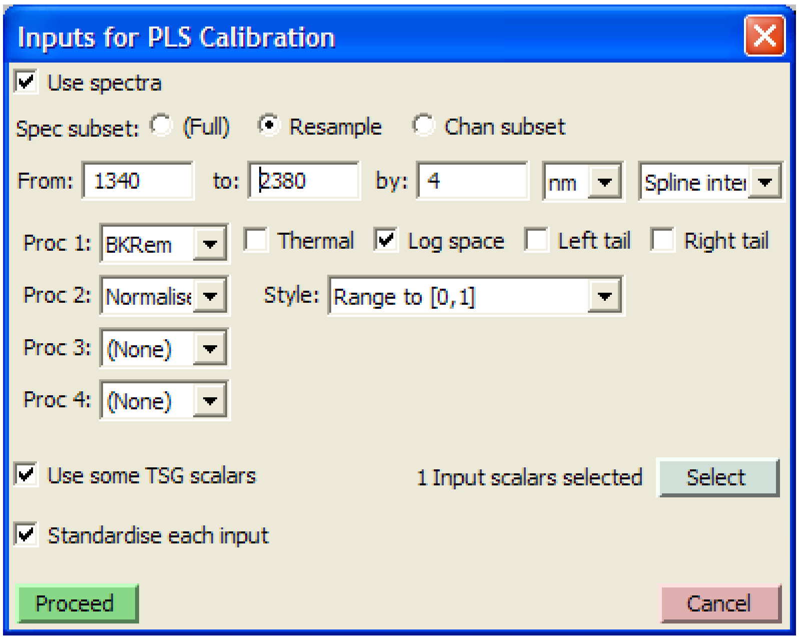
If you have a mask or class scalar that identifies the calibration samples then you can easily use the "selection or deselection by scalar match" tool to select them. It's the same tool as the one in TSG's class editors and I won't describe it here.

A calibration set might be as small as a dozen (risky) or perhaps as large as a thousand or two (excessive). Around 100 is a decent number. Calculations will be *slow* for a large calibration set.

It is common to have fewer calibration samples than spectral channels. (This is not a technical problem for PLS.) It is also common to have highly correlated spectral channels. (This is not a technical problem for PLS either.)

Note: If you make models etc but then change your cal sample selection, you'll have to run CV again.

Inputs



Inputs for PLS Calibration

☑ Use spectra

Spec subset: ○ (Full) ● Resample ○ Chan subset

From: 1340 to: 2380 by: 4 nm Spline inte

Proc 1: BKRem □ Thermal ☑ Log space □ Left tail □ Right tail

Proc 2: Normalise Style: Range to [0,1]

Proc 3: (None)

Proc 4: (None)

☑ Use some TSG scalars 1 Input scalars selected

☑ Standardise each input

The PLS screen takes full reflectance spectra by default. If you want to take a subset of the wavelength range or if you'd like some spectral processing done (e.g., hull quotients) then this is where you do it. Also, there's something I haven't told you yet. You can use some TSG scalars to supplement the spectra (like pseudo spectral channels tacked on), or even forsake spectra altogether and drive the modelling from TSG scalars only.

Note: If you make models etc but then change something in "inputs", you'll have to run CV again.

1.3.4.1 Use spectra

Normally you'd have this on. If you turn it off then you must turn on Use some TSG scalars and proceed to select TSG scalars to drive the model.

1.3.4.2 Spec subset

Use these controls to resample the spectra to a different resolution or to select individual spectral channels. You should have seen this control set in TSG before, e.g., in the Stats module, and I won't give it much attention here. The resampling route lets you take a contiguous spectral subset to a different resolution (normally coarser). The subset route lets you pick individual channels as you please.

Although it looks like you're stuck with a contiguous block of wavelength coverage if you take the resampling route, you will find that you can weed out individual resampled channels later on through the input outlier removal mechanism.

1.3.4.3 Processing

There are four lines of controls for spectral processing. Each one gives a choice of what it should do – background removal, normalisation etc. So you can chain up to four different spectral processing steps together. By default, all four steps are set to do nothing. The PLS screen doesn't give you a choice of spectral layer – it always takes the Reflectance layer – but it gives you this functionality instead.

All spectral processing is done *after* channel subsetting or resampling. (This might change in a future release.)

Well-chosen spectral processing can make a *significant* contribution to a model's success.

PLS strives to find a linear model between the spectra and the concentration and often there's "useless" spectral variability that gets in the way. Removing this variability by spectral processing can help a lot. For SWIR reflectance spectra, hull-quotient background removal or 1st or 2nd derivatives are popular choices.

Sometimes the relationship between the spectra and concentrations is not linear and you may be able to address this in the spectral processing. If the relationship happens to involve multiplicative mixing of spectral "components", taking the logs of the spectra will turn this into additive mixing (which is what PLS likes). If there is a kind-of x^3 thing going on (or the like), taking spectrum$^{1/3}$ (or the like) might undo it. Use your imagination.

You might also consider going in the other direction, working on the concentrations, although you'll have to do it before coming to the PLS screen as there are no tools to do it here. There's nothing stopping you from putting the concentration scalar through TSG's "ARITH" scalar construction method to modify it, before coming to the PLS screen.

1.3.4.3.1 Spectral processing operations

Presently there are seven to choose from, although it looks like there are eight. (I haven't actually implemented "DCT" yet.)

1.3.4.3.1.1 Normalise

Each spectrum gets standardised in some way:

- ZNorm: Shifted to have mean = 0; scaled to have standard deviation = 1.
- Mean to 1: Scaled to have mean = 1.
- Mean to 0: Shifted to have mean = 0.
- Range to [0,1]: Shifted and scaled to have a minimum of 0 and a maximum of 1.

1.3.4.3.1.2 BKRem

Continuum removal is done on each spectrum. Normally a hull-quotient algorithm is applied but if Thermal is checked then one of TSG's lower-continuum-subtraction algorithms is applied instead. You have pretty much the same options here as in TSG's Settings.

1.3.4.3.1.3 Savgol

A Savitsky-Golay filter is applied to each spectrum to perform smoothing and / or calculate a derivative. Conceptually, SAVGOL fits a polynomial to a sliding window that's positioned over each spectral channel, and returns the value of the poly or one of its derivatives. Set Deriv=0 for plain smoothing, 1 for the 1st derivative, 2 for the 2nd derivative and so on. Adjust the poly order if you like. Adjust the window size if you like. (There's a left and a right half so you can set up an asymmetrical window if you want to.) You can consider the amount of smoothing to be related to (left+right+1) / (poly order + 1).

1.3.4.3.1.4 Arithmetic

This option lets you do simple arithmetic with each calibration spectrum and a constant `X` that you type in. E.g., `Spec + X`: each calibration spectrum gets the constant `X` added to each of its channels.

1.3.4.3.1.5 Log

Natural logarithm. Each calibration spectrum is taken to log space. It fails on values of 0 or less. Failed results are set to a large negative value.
It undoes exp, i.e., `log(exp(x)) = x`, assuming exp(x) can be calculated.

1.3.4.3.1.6 Exp

I think this thing's proper name is "natural exponent". It's e^{spectrum}. It can fail on large values. (Don't give it anything over 80 or so.) Failed results are clipped to a high value.
It undoes log, i.e., `exp(log(x)) = x`, assuming log(x) can be calculated.

1.3.4.3.1.7 Power

It raises each calibration spectrum to a power `X` that you type in. E.g., `X=2` gives the square of each spectrum, `X=0.5` gives the square root of each spectrum, and `X=-1` gives 1 / spectrum. In general you should not try a fractional `X` (e.g., 0.3 or 1.5) on negative values, or negative `X` on zero values.

1.3.4.4 Use some TSG scalars

Turn this on if you'd like to supplement the spectral channels with some scalars, or drive the model from scalars only. A button called Select will appear to the right. Click it for a scalar selection dialog.

If you give this a go, you will find only some of your scalars on the list. TSG does not offer any class scalars here, and it does not offer any scalar that contains one or more NULL values (in the samples selected for calibration).

These scalars are on the "input" side of the modelling. Do not confuse them with the single scalar on the "output" side – the "concentration" scalar being modelled.

Before selecting any input scalars, consider what's going to happen later on when you wish to run a PLS prediction on an "unknown" dataset. That other dataset *must* have the same input scalars. This is fine if the scalars were derived from the spectra (you can get them by copy-processing) but you might hit a wall if you imported them from a spreadsheet.

1.3.4.5 Standardise each input

You might have noticed that I'm gradually slipping into this "input" terminology. An input is a spectral channel (after resampling if done) or an input scalar.

Anyway, this option Z-normalises each input – each input winds up with a mean of 0 and a standard deviation of 1 in the calibration set.

"Why, oh why must you make life so darn *complicated*?", you might ask. "First there's spectral processing and now *this*?" This is different. It's mainly for when there are input scalars. It's actually a bad idea, I think, if there are no input scalars, and it's a melancholy compromise if there are spectra plus scalars. It's like this.

If you model just with spectra (no input scalars) then I recommend that you leave this off. There's little action at some wavelengths (e.g., 1700 nm) and lots at others (e.g., 2200nm). That's how things *are*. If you normalise the spectral channels so that each one has the same variability then you'll exaggerate the "quiet" channels (probably meaningless variability) and actually make life more difficult for PLS.

If you throw some input scalars into the mix then unfortunately you should consider doing something to make them blend in with one another and with the spectral channels. It's not *strictly* necessary because PLS is not stupid – it will look for relevant variability in each input even in the presence of scale differences – but it can be beneficial. E.g., If you're using reflectance spectra then channel values will range [0,1], coarsely speaking. If you add a "wavelength" scalar then it might range [2190, 2225], for example, and be shouting in PLS' face: "Me! Pick *ME*, for I am, and none other." PLS will probably cope reasonably well regardless, but some of the evaluation plots will have a crazy spike that'll make them difficult to interpret. If you add a "depth" scalar then it might range [0.0001, 0.01] and have the opposite effect. There is a certain appeal in getting these inputs roughly compatible. "Standardise each input" is the easy, brute-force option, but it is better if you take care of it yourself for each input scalar before doing PLS calibration.

1.3.5 Model this

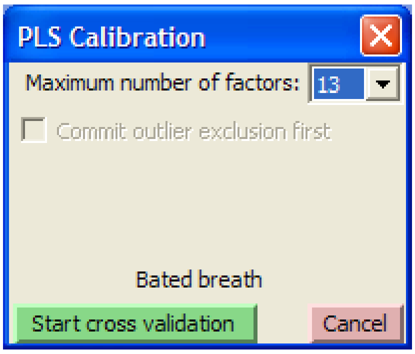
Use this list to select the scalar to model.

You will find no class scalars here.

If you select a scalar that has any NULL values (for samples in the calibration set) then your selection will get rejected.

Note that TSG automatically has a go at making a selection here as soon as you select some calibration samples. In the process, it might first try an unacceptable scalar and give you some strange error message about it before defaulting to an acceptable scalar. Pay it no mind. I'll probably clean it up sometime.

1.3.6 Run CV (cross validation)


PLS Calibration

Maximum number of factors: 13

□ Commit outlier exclusion first

Bated breath

This brings up a dialog for running the CV (cross validation) process, which is central to PLS modelling. It normally has just one setting – the maximum number of factors. It will do cross validation for models of 1 factor, 2 factors, etc up to this number of factors. (Yes, lots of models.) If you have run CV before and are returning after doing some outlier exclusion then you will see another option – Commit outlier exclusion first. You should leave this option on because it's why you returned in the first place.

1.3.6.1 Arms waving in the background

Let's consider a PC (principal components) transform. A PC transform is driven by spectral variability. It takes spectra and transforms them to a different space – "PC space". If you come in with 500 reflectance channels then they get transformed to 500 PC channels but (this is important) only the first 20 or so are worth keeping. Practically all of the dataset's spectral variability has been concentrated into these PC channels. The PC transform has found 20 or so different "things" that matter, and you might as well chuck the rest away. These things are ranked. The first one encapsulates more spectral variability than any of the others, then comes the second, and so on. The PC transform thinks of a spectrum as a mixture of these 20-odd things, but they will probably look artificial to you. You probably won't be going "Ah, thing number 8 is an intermediate chlorite spectrum". They will probably look like mixed up bits of this & that to you. But to the PC statistics, they are the unique and different things associated with spectral variability in the dataset.

PLS is kind-of like a directed PC in that it finds a transform from the spectral domain to "components that matter", in order of importance. While PC is just driven by spectral variability, PLS is driven by a combination of spectral and concentration variability. (PLS models spectral and concentration variability together.) As in a PC transform, almost all of the variability that matters is concentrated in the first handful of PLS channels. A PLS component has a spectrum, like a PC component, but it also has a portion of concentration.

The comparison is starting to get strained round about now. For starters, people don't say "PLS channels" or even "PLS components", they say "PLS *factors*". Next, PLS has two sources of variability that are getting whittled away by each factor. It has the spectral variability *and* the concentration variability. They are getting whittled away in tandem but the one we really care about is the concentration variability. In a PC transform we asked: "How many PC channels are required to describe (almost all of) the spectral variability?", while in PLS our first question is:

1. How many PLS factors are required to describe the concentration variability? (i.e., How many PLS factors must we use before the concentration residual is insignificant, meaning we have modelled the concentration?)

If there are 500 spectral channels then we could go up to 500 factors and nail it completely, but it would be nice if we arrived well before then. So then, how small is small for an "insignificant" concentration residual? Two percent? I know the answer and it is "no". I have led you down the garden path. Our first question seems sensible but it's actually misguided. PLS will normally get the concentration residual as small as you please by using enough factors, but at some point it will start being driven by odd little peculiarities and stupid little bits of noise in the calibration spectra. I'm not making this up; it *will*. At some point it will get so into the calibration spectra that it'll be useless for analysing what's actually going on or predicting a concentration for an "unknown" spectrum that isn't one of the actual spectra in the calibration set. PLS' normal goal is to get a *useful* model, so here's the question we should really ask:

2. What number of factors strikes a nice balance between a robust model and a decent concentration residual?

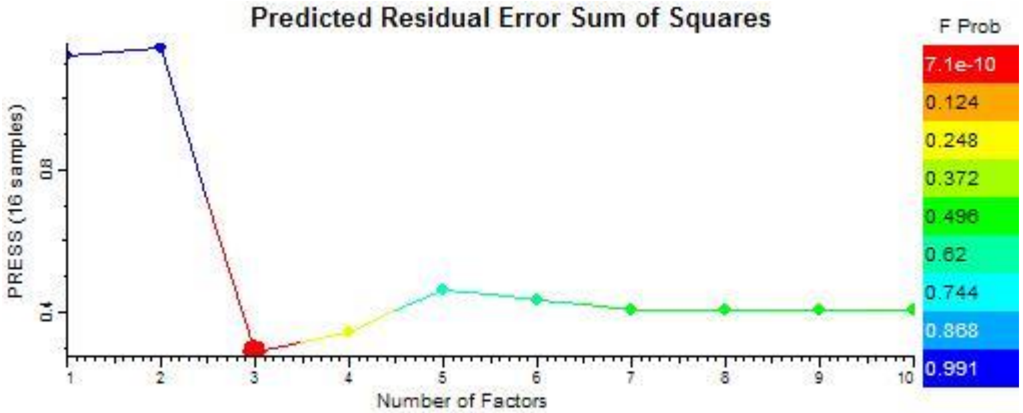
The CV process is quite good at answering this question and the implicit second question too: "How can you tell if a model is robust?"

1.3.6.2 The CV process

It goes like this. Say we have a calibration set of 100 samples and we're going to evaluate models of up to 10 factors. Why 10 factors? I don't know. 10 seems nice. Let's hope it's enough. We'll know soon enough. Anyway...

1. Set up an array called PRESS (predicted residual error sum of squares), with 10 elements (one per factor). Zero the elements.
2. Put one sample aside and make PLS models from the other 99. So that's 10 models – a 1-factor model, a 2-factor model, ...a 10-factor model.
3. That sample we put aside – we know its concentration value. Now *predict* a concentration value for it using each of the 10 models that we just made.
4. Calculate a residual for each of the 10 predictions that we just made. Real value – predicted value. Square each residual and add it to its corresponding PRESS element.
5. Go back to step 2, putting the next sample aside and modelling with the other 99. Stop once we've done this for all 100 samples.

Once the CV process is complete, the PRESS array shows an interesting view of model performance. You typically see it bottom out and then start increasing again with more factors. (In fact, if you *don't* see this then you should run it again with more factors – i.e.,10 is not enough in our example.)



Perhaps you're wondering: "Why should it *ever* bottom out like that? Surely the residual error should steadily get smaller for models with more factors?"[3]

In each of the CV predictions, the sample we're predicting for is an unknown – it is not part of the model's training set. That PRESS minimum is revealing the point (number of factors) where the PLS model is about to get too into the data it was trained on. It is starting to be driven by odd little things that aren't present in the unknown's spectrum and so the prediction is actually getting *worse*, leaving a greater concentration residual. That's why the PRESS bottoms out and starts increasing again.

The number of factors where the PRESS reaches its minimum might already be "too many" but I'll discuss that later, and I'll also discuss some other useful things we get out of the CV process.

1.3.6.3 Maximum number of factors

It is up to you to decide how many factors to evaluate. The whole point is to find the sweet spot – the number of factors where the PRESS bottoms out before increasing again. If you do not see this trend in the PRESS then you should consider increasing the maximum number of factors and running CV again.

Although it is informative to see the PRESS behaviour up to a large number of factors, a consideration is that the higher the maximum number of factors, the longer it will take CV to run. Run time is roughly proportional to [number of calibration samples] times [maximum number of factors]. If you see a PRESS minimum within about a dozen factors, you've probably found that sweet spot. (And in my personal opinion, a model of more than a dozen or so factors is chancy anyway.)

1.3.6.4 Start cross validation

Obviously, you must click this button to make it go. What I really wanted to say here is that you'll see a miniature PRESS plot in the dialog itself as CV progresses, so before long you'll have a clue about whether or not you have specified a high enough maximum number of factors.

[3] Perhaps you're also wondering why the above figure says "18 samples" when I was talking about 100 in my example. Please don't wonder about that. You ought to be thankful for *any* documentation you receive from a programmer. I don't actually have a 100-sample calibration set handy right now. This 18-sample plot shows the trend I wanted you to see.

1.3.7 What now?

So you've selected a calibration set, mucked around with the "inputs" dialog, selected a scalar to model and run CV. TSG has gone and selected what it thinks is the best model (the **Factor** list), and left you staring at the PRESS plot. You will find that there are various options available in the **Plot** list but the PRESS plot is a good place to start. After that, the CV Act:Pred plot will show you if the model is viable. Given a viable model, the FRC plot is probably the best place to look, to find out what's driving the model – what parts of the spectrum matter the most to it. After that it's pretty much over to you. There are other things to look at, and many of the plots support sample or input outlier exclusion – should you wish to try your hand at that.

It's worth mentioning that all sample-based PLS plots (like the CV Act:Pred plot for example) are linked to TSG's "current sample" system. If you have a floater going then it will be updated as you click on samples in one of these plots.

1.3.8 Meet the PRESS plot

See the previous page for an example of an encouraging PRESS plot. Observe that it improves (gets smaller) for the first few factors, then gets worse (gets bigger). This is encouraging because:

- In our work, it is rare to get a decent 1-factor model. The optimal model normally has more than one factor.

  Our spectra are complicated. A mineral usually has several spectral absorption features and, looking the other way, given one particular absorption feature there's often more than one mineral that could cause it. PLS generally needs more than one factor to do its own peculiar brand of unmixing.
- If PLS has actually worked, it is right and proper for the PRESS to bottom out start increasing at some point.

  PRESS is about the model's performance in predicting *unknown* samples. The bottoming-out suggests that PLS found some genuine relationships between the spectra and the concentrations – relationships involving spectral activity that is in both the training set and the unknowns[4]. The worsening at higher factors is *good* to see because we expect PLS to find all the "good stuff" (significant spectral variability) at low factors and be left with odd little bits of junk (in the calibration samples) at high factor levels. We expect there's a point where these odd little bits of junk aren't in the unknown samples any more, making the unknowns' predictions worse.

1.3.8.1 Selecting the final model

Another phrase for this is: "Selecting the number of factors to use". Do it by picking an entry in the **Factor** list.

You should notice that TSG has already had a go. Observe the default selection in the Factor list and the big dot in the PRESS plot. Quite often, you will find that TSG hasn't selected the number of factors where the PRESS bottoms out; it has selected one or even two factors fewer. It might seem like a careless mistake but it's actually a magnificent triumph of statistics. Here's the argument...

If you've been reading this, you should have picked up on some nascent paranoia about a model using too many factors. A typical PRESS plot shows this clearly. It

[4] If you don't know what I mean by "unknowns" here then you should read the CV section again.

bottoms out and gets worse. It gets worse because, at higher factors, the model starts being driven by odd little bits of spectral variability that are in the calibration spectra but not the unknowns.

The question is...

> So the PRESS bottomed out nicely at N factors, but are we in trouble already? Sure, N seems right for CV "unknowns" in this calibration set, but what about when we get round to predicting on *genuine* unknowns, you know, ones measured on some other occasion? I'm *really* worried about dud predictions for genuine unknowns. Maybe we'd be better off using fewer factors?

Then it gets mulled over and expressed more statistically, and now we have some smarty-pants academic reasoning that justifies our nervous desire to use a simpler model at the least excuse...

> I know that a PRESS value is a *sample* of an error, not the *actual* error, so it might be a little bit wrong. I think that the *actual* error for N-1 factors is as good as the one for N factors. What's the probability that I'm wrong?

In statistics there's a thing called the F test for determining probabilities like this. It involves some assumptions so I wouldn't call it bullet-proof, but it's useful. It will say something like: "Given those two PRESS *samples*, the probability that the first *actual*, underlying error is bigger than the second one is 0.1."

Well that's interesting, I suppose, and downright civilised of the F test to give me one number in exchange for another, but what do you imagine a good probability cutoff might be? People who are into PLS and have tried stuff out suggest that `0.125` is a useful probability cutoff, in practice, for deciding whether or not PRESS[N-1] really is bigger than PRESS[N]. This is the cutoff that TSG uses to select the default final model.

1.3.8.2 Wrapping up the PRESS plot

The main use of the PRESS plot is as a guide to selecting the final model, and I've just told you about that. The plot is coloured by F probability. I've told you a bit about that and now I'll tell you some more. If you click on the plot to get a cursor readout, you'll see there's another thing called SEP. I'll tell you about that too.

1.3.8.2.1 PRESS F probability

There's an F probability for each factor level. Each one is relative to the smallest PRESS in the plot. Let's say the smallest PRESS is at N factors. The F probability for M factors (M != N) means: "This is the probability that the *real* error at M factors is bigger than the *real* error at N factors". For a probability, 0 means "certainly not" and 1 means "certainly so".

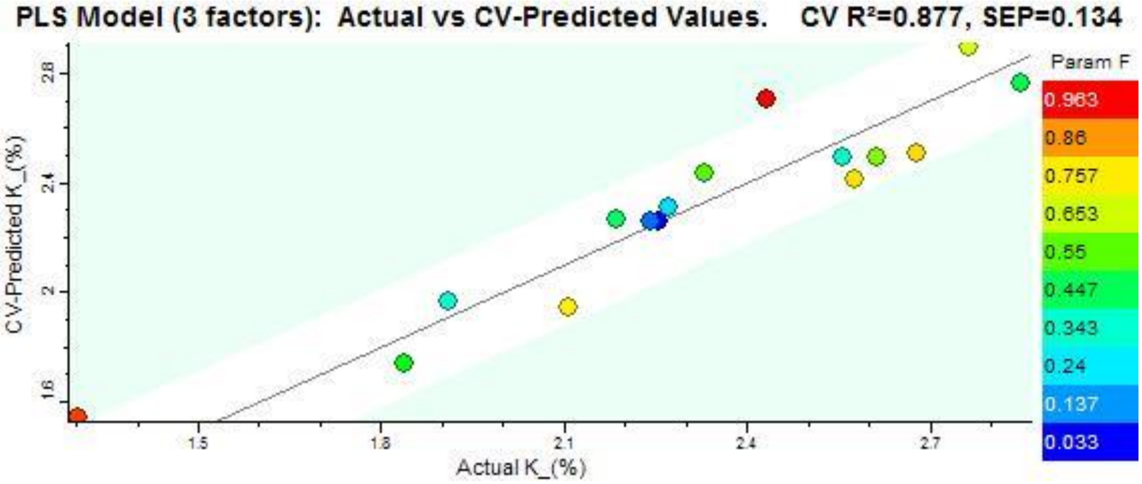
The way F probability is used in TSG's PRESS plot is: "this is probably bigger than that". You might come across different F probabilities in the literature. If you come across big numbers like 0.8 or 0.9, they might mean "this is probably the same as that", or perhaps "this is probably the same or smaller than that". If you come across numbers like 0.25, they might mean "this is probably not the same as that (this is bigger or smaller than that)". You have to watch out for subtleties when comparing F probabilities.

1.3.8.2.2 SEP (Standard Error of Prediction)

For a factor level N, SEP[N] is simply SQRT(PRESS[N] / number_of_samples). In other words, SEP[N] is the RMS error for the N-factor model. (Some people call it RMSEP.) It's an average error for the model, in the same units as the "concentration" – the thing being modelled. It is based on CV prediction errors, where each calibration sample is predicted as an unknown. It represents the average error you might expect when predicting for genuine unknowns, provided these unknowns are "like" the calibration samples.

1.3.8.3 Data export

If you export the PRESS plot to a CSV file or to the clipboard then you get the following for each factor level that was calculated: PRESS, SEP and F probability.



1.3.9 The CV Act:pred plot

You'd be hard-pressed to find a better indicator of model performance than this. It shows how good the model is at predicting unknowns. It scatters known concentrations along X and predicted concentrations along Y. (These predictions come from the CV process where each sample, in turn, was predicted as an unknown.) There's a thin grey line drawn for Y=X to show what perfection would be.

The plot title shows the model's "R squared", which (in this case) is the "coefficient of determination" and it'll get a mention later. It also shows the model's SEP, or Standard Error of Prediction. It's the model's RMS error, in the same units as the concentration. A recent addition (not shown in this graphic's title) is the model's "Bias", which is the average of actual-predicted[5].

If you "mouse" around in the plot with the left button down, you'll see the nearest sample get highlighted in the plot and in the list. You'll also get a cursor readout with the sample's actual and predicted concentrations and its two residual Fs.

If you left-click a sample on the list then you'll see the sample's dot get highlighted in the plot. You can also <CTRL>click to toggle samples, or <SHFT>click to select a block of samples. (Several samples can be selected, not just one.) Later on I'll tell you *why* you might want to select samples.

This plot can drive a floater link. If you have one or two floaters going then they will display their stuff for the current sample as you mouse around in this plot.

[5] Purists call *my* SEP the Root Mean Squared Error of Prediction, or RMSEP, and define another fussier SEP such that RMSEP^2 = SEP^2 + bias^2.

1.3.9.1 Residual effs

Each point is coloured by the F probability that its residual is "bigger than normal". There are two kinds of residual to choose from – the parameter (or concentration) residual and the spectral residual. Make your choice with the **Colour** list.

The green-shaded regions show where you (yes, you) think the parameter residual F probabilities are significant. (*It's always the parameter F*, even if you are colouring by the spectral F.) Even from a distance I can hear the sounds of protest but I insist the choice is yours. There's a field called **Param F** above the plot and that's where *you* type in *your* cutoff. It defaults to `0.9`, which is a cutoff I've seen recommended in literature, but don't just take it as "the rule". Give it some attention.

1.3.9.1.1 More about the effs

As noted, there are two kinds of residual in a CV prediction of an unknown.

First there's the parameter (or concentration) residual. CV predicts a concentration value for the "unknown" spectrum. Although the sample is an "unknown" to CV, it is actually from the calibration set, therefore we have its true concentration at hand. The residual is [true concentration – CV-predicted concentration]. We square it to work with the F stat.

Then there's the spectral residual. The PLS prediction takes in an unknown spectrum and leaves behind a residual spectrum. In this context, the "spectral residual" is the sum of squares of this residual spectrum. (It's a single number per sample.)

Either way, we get an average residual for the whole calibration set, then get an F probability for each sample. In this context, the F probability means:

> **This sample's *real* residual is probably bigger than average.**

The first level of abstraction takes this to:

> This sample's residual is probably too big.

The second level takes it to:

> There's probably something wrong with this sample.

But *watch out* for abstraction. I'll have more to say about this later. For now, I'll just say that there's a level 2 ½:

> ...or maybe there's something wrong with this model?

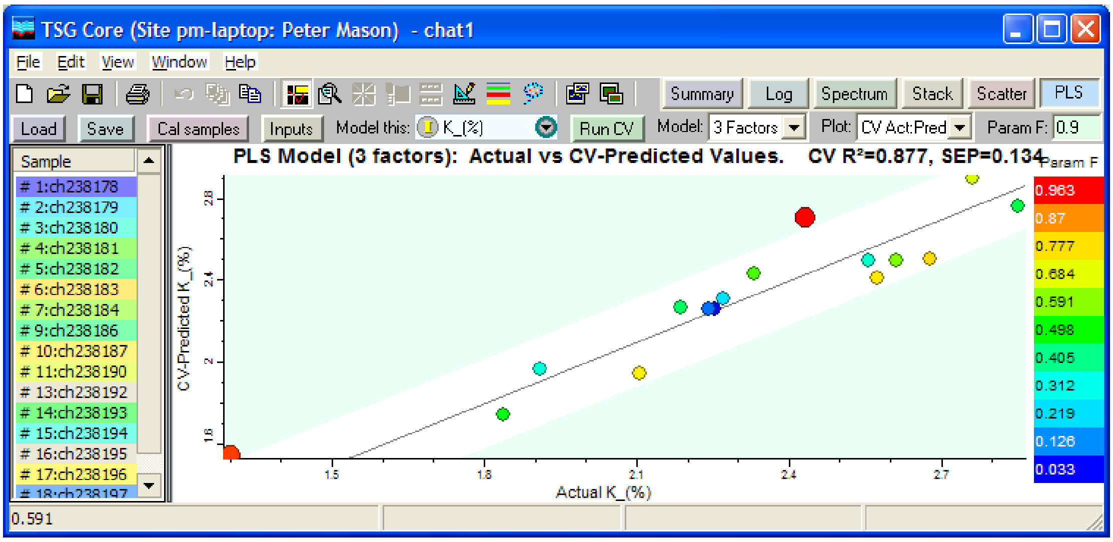
The spectral residual F is not offered in this plot as an outlier identification method; it is only offered as a side interest, for colouring the dots. In PLS calibration, spectral residual is a poor second cousin to parameter residual. The CV Act:Pred plot is important and I didn't want to confuse it with feeble functionality. If you really want, you can muck around with a spectral F threshold in a later plot.

1.3.9.1.1.1 Why Spec F is a poor second cousin to Param F

I've told you that in PLS modelling, the spectral and concentration residuals are reduced in tandem with each new factor, but the one we really care about is the concentration (param) residual. If the concentration residual gets reduced to insignificance then we know that we have modelled it well. We don't really care what the spectral residual is at this stage. In my opinion it is perhaps a little encouraging if there is a noticeable spectral residual, because it shows that PLS did not require all of the spectral variability to model the thing we want – the concentration. It has fished out just the spectral variability that matters in modelling the concentration, and it has coped in the presence of other spectral variability.

It is likely that the different calibration samples have different bits of so-called junk left behind in their spectral residuals. There is nothing wrong with this. If one sample has a greater overall spectral residual than the others then it is not necessarily a bad sample. Maybe it just has more “irrelevant” spectral variability. It might even be a very useful sample, because it might have helped more than usual to “teach” PLS how to pull out spectral variability that matters to the modelling from variability that doesn’t. You might be thankful for this sample later on, when you run predictions on *real* unknowns.

1.3.9.2 Outliers and the list, and some other things too



Model this: K_(%)

Model: 3 Factors

Plot: CV Act:Pred

Param F: 0.9

The CV Act:Pred plot is accompanied by a list showing all of the calibration samples. The list items are coloured by Spec F or Param F (whichever one you have selected for plot colouring). The first time you see this plot, you might notice that each and every sample-name in the list is preceded by a hash (#) symbol. If you **right-click on the list**, you’ll get a special menu of special things to do with the list.

I told you earlier about how you can select samples in the list, and also select a single sample by left-clicking in the plot. Now cast your gaze to the row of toolbar buttons in the graphic above. You should see a little blue lasso thing, like this: If you click it, you will be able to select a whole bunch of dots in the plot by lassoing them. And now the time has come to talk about *sample* outlier removal.

1.3.9.2.1 Sample outliers

There are two kinds of outlier removal in TSG’s PLS calibration – *sample* and *input* outlier removal. This is the first of three plots that deal with sample outliers. Later on I’ll get to the other plots and the input outliers.

So then, abrasive Q & A time once again.

- What’s a sample outlier?

  Two things, both mistakes:

  A sample where a *mistake* has been made in the spectral or concentration measurement.;

  A sample that’s genuinely foreign. A *mistake* was made in its selection, or someone’s been chucking rocks around the place, or something. The sample does not originate from the same environment as the other samples.
- What’s *not* a sample outlier?

  Anything else, including appropriate samples with good measurements that give big F residuals in your stupid model.
- Well, uh, o-kay, then how the can I use this plot to select outliers?

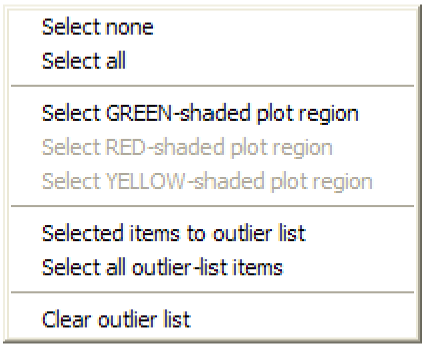
  You mustn’t just go for it. Use this plot as a *guide*. If a mistake has been made with a sample’s measurement or selection then it’ll probably show up here with a high param residual F. So look for samples with high param Fs, then *check* each of those samples to see if any mistakes have been made, and finally mark the ones with mistakes as outliers.

- Forget that. I'm just going to chuck out all samples with big param Fs.
  Knock yourself out. Just don't come crying when your model's predictions are rubbish.
- Well, what should I do with these high-F samples then?
  If a sample is appropriate and its measurements are good but it gives a big param F, you have to deal with it. It *came* from the environment you're interested in, right? You are going to see others like it in prediction later on. The simplest thing to do is just to keep it. It is probably contributing usefully to the model – making the model a bit smarter about the weird spectral variability it contains. You'll just have to take the hit that predictions on "unknown" samples like it will have comparatively big errors. There is a measure (spectral residual) that might catch such samples in prediction, but it isn't that great. I don't think you can count on it as much as you'd like. A better course of action is to work on the model – try some other spectral processing or something. Another thing to try is to add more calibration samples, similar to the one that's currently giving a big param F. Or try this... Go ahead and exclude that sample from calibration, but try to identify what's peculiar about its spectrum so that you can devise a masking strategy. Later on when it comes to predicting on unknowns, apply the mask, saying: "the model can't be used on samples like this". Taking that last idea further, you might simply require two or more models for this problem. The variability might be too complicated for one model, or there might be "more than one thing going on" in the relationship between the spectra and the concentrations. This approach is harder, though, because you will have to come up with some classification system to partition your calibration samples and (later on) your unknown spectra.
- How big is big for a param F anyway?
  In the literature I've seen probabilities like `0.9`. So, pretty high. The interpretation is: "This sample's *real* concentration residual is *most probably* bigger than normal."
- In the PRESS plot you had an F probability cutoff of 0.125 but here you come along with 0.9. I'm confused.
  I like it when you're confused. It feeds my vanity. I'll let you in on this one though. In the press plot, the test was, in essence: "This model is probably worse than the optimal model". We were conservative with the probability cutoff because we wanted a model that was as good as the optimal one (but used fewer factors). In the CV Act:Pred plot the test is, in essence: "This sample is probably an outlier". We use a high cutoff here because we are very reluctant to discard a sample as an outlier. We want the F probability to be almost certain of it before we consider it.

1.3.9.2.1.1 Marking samples as outliers

It's a sad, sad moment when a precious calibration sample is marked as an outlier but sometimes it has to be done. I've told you how to select samples in this plot. Selected samples can be marked as outliers. This is done using the list's own right-click menu or, if you insist, the plot's right-click menu (List actions tree) or TSG's main menu (View -> PLS -> List actions tree).

The first thing to know is that **the list has two levels of selection**. You've see the first. Selected items are dark blue (or grey) in the list and they flash in the plot. Things are volatile at this level. Beneath this is a second level – the more permanent outlier-marking level. It is harder to get at and change. You can only get at it through the menus. Selection status at this level is shown by the hash (#) symbols in the list. A hash means the item is active and a space (no hash) means it has been marked as an outlier.


Select none
Select all
Select GREEN-shaded plot region
Select RED-shaded plot region
Select YELLOW-shaded plot region
Selected items to outlier list
Select all outlier-list items
Clear outlier list

Okay, let's move on to that menu.

The first five items are yet more ways to mess around with the volatile first-level selection. Select green-shaded plot region is worth a mention. (The other shaded-region items are greyed out because this plot only has a green-shaded region. Other plots have other shaded regions.) Remember that you can control the green-shaded region in this plot by typing in a Param F cutoff? This is a quick way to select all samples with a param F above this cutoff.

- Selected items to outlier list
  Anything selected at the first level will get marked as an outlier at the second level. You will see its # disappear. It's a cumulative action. Things that were marked as outliers before will be left alone – they will remain marked as outliers. The only change that can happen is that some new items get marked as outliers.
- Select all outlier-list items
  This copies the second level onto the first. Anything without a # will be blue in the list and flashing in the plot.
- Clear outlier list
  This resets the second level. Every item has a #. No outliers are marked.

1.3.9.2.2 Committing outlier exclusion

You don't have to commit outlier exclusion right after marking outliers. You can fiddle around with other plots and with the other type of outlier (input outliers) as you please, until you're ready.

However, the moment you mark anything as an outlier, you'll see the **Run CV** button go red. This means: "You have changed something that affects the model. You ought to run CV again to update the model, sometime soon before you get too carried away with what you've got now." So when you're ready, click that Run CV button. You will find that there's a checkbox called Commit outlier exclusion first in the CV control panel. Leave it on and start the CV process to update the model. If you turn it off instead then your outlier exclusion will be abandoned.

1.3.9.2.2.1 Changing your mind

If you commit sample outlier exclusion and regret it later, you'll have to reinstate the samples yourself. Click Cal samples and go right back to the selection of the calibration set.

A safer way to travel is to save your PLS session before doing something major like committing sample outlier exclusion. (You can have as many saved-session files as you like.) It is quick and easy to restore a saved session.

1.3.9.3 R squared?

This plot shows a thing called R^2 in the title. Normally in a TSG plot I'd say R^2 is (the Pearson's correlation coefficient) squared but in the PLS screen I use a different phrase for R^2: "**the coefficient of determination**". (I think it is actually the same number as Pearson's correlation squared but let's not pop that bubble.) It is a sum-of-squares ratio thing that looks like this:

$$1 - [\sum_i(act_i - pred_i)^2 \;/\; \sum_i(act_i - \mu_{act})^2]$$

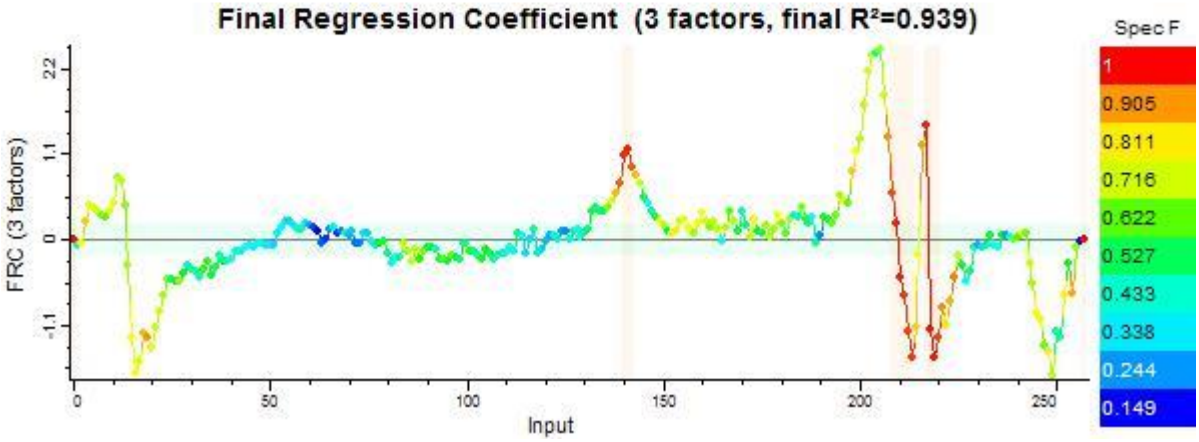
..where `act` and `pred` are the actual and predicted concentrations (arrays), μ_{act} is the average actual concentration, and the sums loop "i" over the number of calibration samples.

Note that in the CV Act:Pred plot, R^2 is derived from the CV predictions where each sample, in turn, is predicted as an unknown. If you want to quote an R^2, *this* is the one to quote. You will see another (normally higher) R^2 in the next plot (the FRC plot). That one's biased, because it comes from the *final* model that "knows" *all* of the calibration samples. I put it in as a matter of interest. You would be cheating a bit if you quoted that final one (instead of this CV one) in relation to model performance.

1.3.9.4 Data export

If you export the CV Act:Pred plot to a CSV file or to the clipboard then you get the following for each calibration sample: sample name, actual concentration, CV-predicted concentration, param F, spec F.

1.3.10 The FRC plot



Here it is, the mysterious FRC, or "final regression coefficient". It seems that not many people know about it. Inge Helland knows about it and describes how to get it[6]. I know about it too, for it would appear that I am the one who read Helland's paper.

[6] I.S. Helland, Commun. Statist. -Simula, 17(2), 1988, pp581-607: "On the structure of Partial Least Squares Regression".

- The FRC is the **m** in $y=\mathbf{m}(\mathbf{x}-\mathbf{c})$, which is almost[7] the complete PLS prediction formula. It is the gain component of PLS' linear model. It is the *heart* of the model, for the offset component (**c**) is just the average spectrum of the calibration set.
- The FRC shows the inputs (or spectral channels if you prefer) that matter the most to the PLS model. Being an array of gains, any place where the FRC is strongly positive or negative is a place that matters a lot to the model. Basic interpretation is simple. A positive FRC value means that input adds to the prediction and a negative value means it subtracts. Proper interpretation can be more complex.
- The plot title shows a "final R^2" value. This is a coefficient of determination, as in the CV Act:Pred plot, but it is calculated on different predictions. The final model "knows" *all* of the calibration samples so its predictions are a little better than the CV ones. The final R^2 is normally higher than the one shown in the CV Act:PRed plot.
- The plot has a green-shaded region governed by a threshold called **FRC min**. It applies to the absolute value of the FRC and is used to highlight inputs that don't contribute much to the model. The default FRC min threshold is *quite arbitrary* and you shouldn't just go with it.
- The plot is coloured by a kind of spectral residual. In addition there is a red-shaded region governed by a threshold called **Spec F**, relating to this spectral residual. It is used to highlight inputs that have large spectral residuals (through the calibration set). Again, don't just go with the default threshold.
- The list shows calibration inputs (not samples), coloured by spec F. It has its own right-click menu with options to select the green, red or yellow-shaded plot regions. (Yellow = both green + red together.)
- If you click on the plot, you get told the nearest input's name (e.g., a wavelength for a spectral channel), its FRC value and its spec F.
- The FRC plot also supports lasso-mode selection.
- Selection in the FRC plot applies to *input* outliers, but I use the term "outliers" rather loosely. There's an outside chance that you *might* use it for input "outlier" removal (inputs with large spectral residuals) but mostly it's for removing inputs that don't contribute much, in order to simplify the model.

1.3.10.1 Interpreting the FRC

I repeat: The FRC is an array of gains for the linear PLS model, where a positive FRC value means that input adds to the prediction and a negative value means the input subtracts. Direct interpretation is obvious.

Actual interpretation requires a knowledge of mineral spectra. We often deal with spectral *absorptions*, which are negative after mean subtraction and so essentially invert what I typed about FRC signs. Moving on, robust familiarity with mineral spectra is necessary for discovering the mineralogy that's actually driving the PLS model. E.g., "kaolinite drives the prediction up while chlorite counter-balances and

[7] If input standardisation is not used then the complete fast-PLS-prediction formula used in practice is: $y=\sum_i[\mathbf{m}_i*(\mathbf{x}_i-\mathbf{c}_i)]+d$, where **m** is the FRC, **c** is the mean calibration spectrum, d is the mean calibration concentration, **x** is an unknown spectrum and y is the unknown's concentration prediction. If input standardisation is used then the formula is: $y=(\sum_i[\mathbf{m}_i*\mathbf{x}_i])*e+f$, where e and f are a gain and offset (respectively) that undo the Z-normalisation that was (also) done on the modelled concentration. The sums loop i over the inputs.

drives it down". An upside-down absorption feature and its implied "double negative" shouldn't phase you. In my opinion, mineralogical interpretation is one of the most valuable things you'll get out of PLS.

Something that's a joy to behold in the FRC is a strong positive spike followed closely by a strong negative one, or vice-versa – a see-saw like arrangement. Such an arrangement often means that the PLS model is being driven by the *position* of a spectral absorption (or peak), or possibly the *slope* of one side of an absorption (or peak). It is a joy to behold because our spectrometers measure absorption positions and slopes relatively reliably, and we don't have to worry much about albedo differences (whatever their causes). Changes in absorption position are often related to changes in mineral chemistry, so it's a cool thing to behold too.

1.3.10.2 Input exclusion

1.3.10.2.1 FRC magnitude

This is mostly what "input exclusion" is about. Inputs with relatively small positive or negative FRC values don't contribute much to the model. If you wish to simplify the model (often a good thing) then these inputs are likely candidates for exclusion.

The **FRC min** field allows you to specify a minimum cutoff. Any input whose absolute FRC value is smaller than this will get a green background, and can be selected by the list's "Select green-shaded plot region" menu. Selected items can be marked as outliers by the list's "Selected items to outlier list" menu.

The default FRC min threshold is *quite arbitrary*. There is nothing orthodox about it and you should not just go with it.

1.3.10.2.2 Spectral residual F

Unlike other plots where spectral residual F probability is calculated for each *sample*, this one's calculated for each *input*, e.g., each spectral channel.

The **Spec F** field allows you to specify a cutoff (maximum allowed) spectral residual F probability. Any inputs with a spec F above this get a red background, and they can be selected (if *really* necessary) by the list's "Select red-shaded plot region" menu.

The default Spec F threshold is quite arbitrary too, and you should not just go with it.

Although there's this mechanism to exclude spec F outliers, I don't expect it to see much use in practice – at least, not on its own. The interpretation of this measure is: "The *real* residual for this input is probably bigger than the average for all inputs." And the interpretation of that interpretation is: "...um, so what?"

Well at least it's a cool way to colour the FRC plot.

I told you before that I think it's a *good* thing for PLS calibration to leave a noticeable spectral residual behind. It suggests that the PLS model is smart enough to pick out just the spectral variability that matters to the model. However, in my opinion it is difficult to interpret why one *input* has a relatively high spectral residual without looking at other plots, and even then – good luck. If you want to know the relative *importance* of that input, just look to the magnitude of the FRC.

I suggest that an ideal input to mark for exclusion is one with a small |FRC| value and a high residual F. It would be in a **yellow**-shaded plot region. The small |FRC| value says "this input doesn't contribute much to the model" and the high residual *suggests* "relatively little of the input's overall variability was used".

I'll wrap up the FRC plot by pondering the meaning of an input that has both a large |FRC| value and a large spectral residual. The large |FRC| plainly says: "This input is important to the model". If the input is important then why wasn't its spectral variability "used up"? Why the relatively large residual? Frankly, I don't have an answer, or not presently at least. I just have a suspicion. I see this situation sometimes around FRC "see saws" and the like. So a large |FRC| with a large spectral residual suggests to me that the model is doing something more sophisticated (less direct) than just responding to an absorption depth. It is responding to an absorption position, sharpness, shape, shoulder slope, or something like that.

1.3.10.3 Data export

If you export the FRC plot to a CSV file or to the clipboard then you get the following for each input: input wavelength or name, calibration input (spec) average, FRC, spec F. In addition, the FRC column-header text includes the calibration concentration (param) average, or the gain & offset used to rescale the predictions if input standardisation was done.

If you like, you might be able do fast predictions yourself, externally (not in TSG), using this information. Normally the formula is:

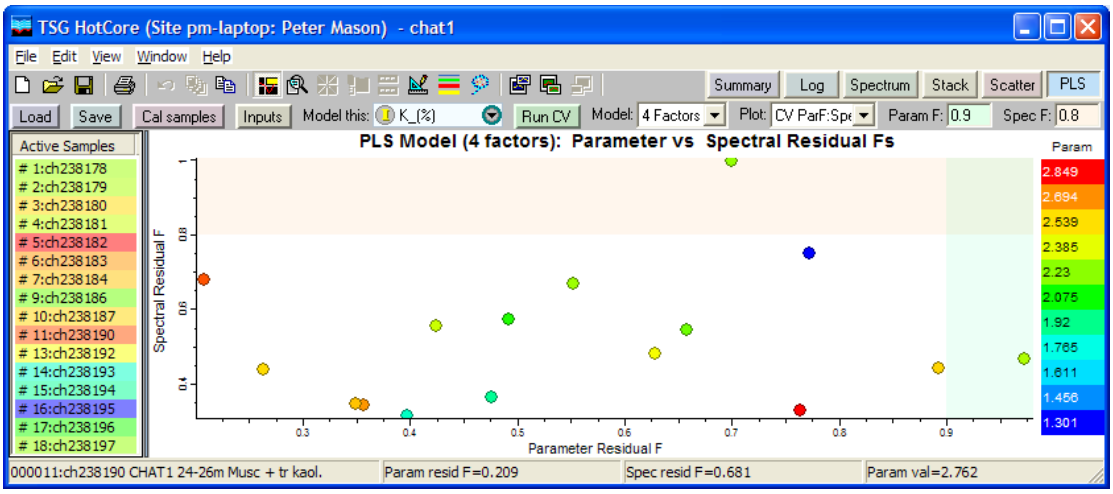
Prediction = sum[(unknown_spectrum[i] – spec_avg[i]) * FRC[i]] + param_avg

If input standardisation was done then the formula is:

Prediction = sum[(unknown_spectrum[i] – spec_avg[i]) * FRC[i]] * gain + offset

The sums loop 'i" over the inputs.

If you intend to do external predictions then bear in mind that unknown spectra must receive the same spectral processing and channel subsetting / resampling that the calibration spectra did.


TSG HotCore (Site pm-laptop: Peter Mason) - chat1

Model this: K_(%)
Model: 4 Factors
Plot: CV ParF:Spe
Param F: 0.9
Spec F: 0.8

000011:ch238190 CHAT1 24-26m Musc + tr kaol.
Param resid F=0.209
Spec resid F=0.681
Param val=2.762

1.3.11 The CV ParF:SpecF plot

I have introduced you to the spectral and parameter (or concentration) residual Fs already, and bored you about their thresholds. The CV process yields these residual Fs for each calibration sample and here we have the two scattered against one another. The parameter residual F drives X and the spectral one drives Y. The **green**-shaded region of the plot is the "high param resid F" zone and it is controlled by the threshold typed into the **Param F** on-screen control. The **red**-shaded region is the "high spec resid F" zone and it is controlled by the **Spec F** on-screen control. The combined region, which I like to think of as **yellow**, is the zone where both residuals are high. The list items (one per calibration sample) and the scatterplot points themselves are coloured by actual concentration values.

As usual, you can mouse around in the plot, highlighting sample points and their corresponding list items. On LMB-down you'll get a readout of the two residual Fs and the concentration value for the selected sample. If you have a floater going then it will get updated as you change the current sample.

This plot provides another way to mark sample outliers and in this respect it has a lot in common with the CV Act:Pred plot. You can lasso groups of points in the plot, select many items in the list, change thresholds (two this time) to change the shaded regions, and use the list's right-click menu to do quick selection actions (three shaded regions this time) and manage sample outliers.

Remember what I said earlier, though. The parameter residual F is the item of choice for outlier detection; spectral residual F is a poor second cousin. That said, I guess it's safe to add that a sample plotted in the yellow zone is more suspicious than one in the green zone.

1.3.11.1 Data export

If you export the CV Parf:Specf plot to a CSV file or to the clipboard then you get the following for each calibration sample: sample name, param F, spec F, actual concentration.

1.3.12 Weights, scores, loadings? The PLS algorithm

Before I describe any other PLS items, I'll have to present the PLS1 algorithm.

TSG uses the "PLS1" algorithm, which deals with one concentration (as opposed to the "PLS2" algorithm's many concentrations). There are two basic kinds of PLS1 algorithm and TSG uses the kind that gives orthogonal scores (as opposed to orthogonal loadings). It goes like this...

We come in with a matrix $\mathbf{X}$ of spectra or inputs and an array $\mathbf{y}$ of concentrations. $\mathbf{X}$ has ns rows and ni columns, and $\mathbf{y}$ has ns rows. (There are ns calibration samples and ni inputs.) Each spectrum (row) of $\mathbf{X}$ has had the calibration mean spectrum subtracted from it, and each element of $\mathbf{y}$ has had the calibration mean concentration subtracted from it.

What now? We realise that we can get something done if we consider the spectra and concentrations as one combined dataset, and break it down into components, or factors. Each factor will have a spectrum and a bit of concentration. Each sample in the combined dataset (spectra and concentrations) will be modelled by a set of scores times these factors. So this is what we set out to do and we use PLS for the job. Once we have these factors, we will be able to set things up so that we can get a concentration when we just have a spectrum (PLS prediction).

The PLS algorithm iteratively factorises $\mathbf{X}$ and $\mathbf{y}$, finding loadings (or factors) $\mathbf{p}_i$ & q_i, scores $\mathbf{t}_i$, and leaving residuals $\mathbf{E}_i$ & $\mathbf{f}_i$. If we calculate m factors, we get the decomposition:

$$\mathbf{X} = \mathbf{t}_1\mathbf{p}_1' + \mathbf{t}_2\mathbf{p}_2' + \ldots + \mathbf{t}_m\mathbf{p}_m' + \mathbf{E}_m$$
$$\mathbf{y} = \mathbf{t}_1 q_1 + \mathbf{t}_2 q_2 + \ldots + \mathbf{t}_m q_m + \mathbf{f}_m$$

Where $\mathbf{E}_m$ and $\mathbf{f}_m$ are spectral and concentration residuals after m factors. M is typically the number of factors in the final model. Note that the same set of scores ($\mathbf{t}_i$) is used in both the spectral ($\mathbf{X}$) and concentration ($\mathbf{y}$) factorisations.

- To kick off, we set $\mathbf{E}_0=\mathbf{X}$ and $\mathbf{f}_0=\mathbf{y}$. That is, each of these residuals starts life as "the whole thing".

Now we find factors $i=1$ to m:

- Calculate weight vector $\mathbf{w}_i = \mathbf{E}_{i-1}' \# \mathbf{f}_{i-1}$, then divide $\mathbf{w}_i$ by its sum of squares to normalise it
- Calculate score vector $\mathbf{t}_i = \mathbf{E}_{i-1} \# \mathbf{w}_i$
- Minimise $\mathbf{E}_i = \mathbf{E}_{i-1} - \mathbf{t}_i\mathbf{p}_i'$ using least squares, solving for $\mathbf{p}_i$
- Minimise $\mathbf{f}_i = \mathbf{f}_{i-1} - \mathbf{t}_i q_i$ using least squares, solving for q_i

Here, weight $\mathbf{w}_i$ is a vector of `ni` elements, score $\mathbf{t}_i$ is a vector of `ns` elements, loading $\mathbf{p}_i$ is a vector of `ni` elements, loading q_i is a scalar, residual $\mathbf{E}_i$ is a matrix of `ns` rows by `ni` columns, and residual $\mathbf{f}_i$ is a vector of `ns` elements. All vectors are column vectors and the apostrophe indicates a transpose. (e.g., $\mathbf{p}_i'$ is a row vector.)

You've seen something about $\mathbf{f}_i$[8] before. $\mathbf{f}_i$ is the array of concentration residuals at factor level `i`. PRESS, SEP and "param residual F" are calculated from it (or something analogous to it, assembled during cross validation). It's a key performance indicator for PLS.

You've also seen something about $\mathbf{E}_i$ before. $\mathbf{E}_i$ is the matrix of input or spectral residuals at factor level `i`. You've seen a "spectral residual F" vector derived from summing squares down the rows (FRC plot) or across columns (Act vs Pred plot) of $\mathbf{E}_i$ (or something analogous to it, assembled during cross validation). Soon you'll see more "spectral residual" vectors derived from $\mathbf{E}_i$.

You haven't seen any of the other items yet (although FRC_i is derived from $\mathbf{w}_j$, $\mathbf{p}_j$ and q_j, $j=1..i$ [9]).

1.3.12.1 Loadings

You get two loadings at each factor level `i`: input or spectral loading $\mathbf{p}_i$ and concentration loading q_i. PLS has broken both the inputs (spectra) and concentrations down into these "pure" components.

Q_i is the concentration contribution of the `i`th factor. TSG doesn't give you a plot of it but you can see it in "loading" item titles, in the "model items" plot which is discussed later on. I don't think you can use it as simply as you might like because the factorisation involves scores too, but there it is.

$\mathbf{P}_i$ is the spectrum of the ith factor – the "pure thing" that matters most in modelling what's left of the input and concentration variability at factor level `i`. The first factor is the most important, then the second, and so on.

PLS' notion of a "pure thing" will almost certainly be different to yours, but for early factors (especially the first factor) there'll normally be some recognisable spectral structure in $\mathbf{p}_i$. Occasionally you might get an early $\mathbf{p}_i$ where the whole thing looks like a single mineral spectrum but more often it'll show a collection of different mineral features, some of which might be upside-down compared to the others.

$\mathbf{P}_i$ becomes more abstract at higher factor levels as the lower factors normally take out most the "good" variability, leaving little for the higher factors to chew on.

The overall sign of $\mathbf{p}_i$ is arbitrary.

1.3.12.2 Scores

I''ll repeat:

$$\mathbf{X} = \mathbf{t}_1\mathbf{p}_1' + \mathbf{t}_2\mathbf{p}_2' + \ldots + \mathbf{t}_m\mathbf{p}_m' + \mathbf{E}_m$$

[8] I'm kind-of regretting calling it $\mathbf{f}_i$ because I don't want you to confuse it with F probabilities. But you wouldn't do that, would you?

[9] $\mathbf{FRC}_i = \mathbf{W} \# (\mathbf{P}' \# \mathbf{W})^{-1} \# \mathbf{Q}$, where $\mathbf{W}$ is the matrix of weight vectors $\mathbf{w}_1..\mathbf{w}_i$, $\mathbf{P}$ is the matrix of spectral loading vectors $\mathbf{p}_1..\mathbf{p}_i$, and $\mathbf{Q}$ is the vector of concentration loadings $(q_1..q_i)'$

i.e., PLS describes each calibration spectrum as a linear mixture of the loadings $\mathbf{p}_i$ (and a residual).

I like to think of the scores $\mathbf{t}_i$ as “the PLS transform” of the calibration spectra. They are like what you get when you put spectra through a forward PC or MNF transform. As such, scatterplots of one score versus another can be interesting. They can reveal groupings and outliers.

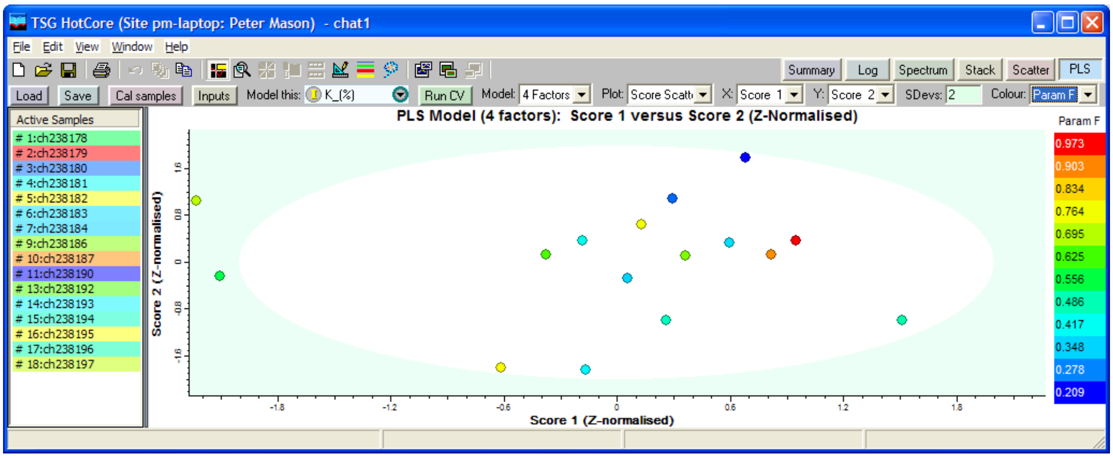
1.3.12.3 Weights

What insights can come from looking at plots of weight vectors? I’m struggling here, I have to tell you. Still, I’ll have a go.

In the algorithm description above, it is interesting how the weight vector $\mathbf{w}_i$ is derived from both the input (spectral) and concentration residuals ($\mathbf{w}_i = \mathbf{E}_{i-1}' \# \mathbf{f}_{i-1}$). $\mathbf{W}_i$ is somewhat like a bunch of correlation coefficients between each input (spectral channel) of the residual spectra $\mathbf{E}_{i-1}$, and the residual concentrations $\mathbf{f}_{i-1}$. My crude take is that *this* is the essence of how PLS derives a transform based on spectral *and* concentration variability, distinguishing itself from the PC transform which is driven only by spectral variability. Also, one might say that $\mathbf{w}_i$ is a forward transform vector of sorts ($\mathbf{t}_i = \mathbf{E}_{i-1} \# \mathbf{w}_i$) and I suppose that one might analyse it in that context. If this looks exciting to you then take another look. $\mathbf{E}_{i-1}\#\mathbf{w}_i$. The very first weight $\mathbf{w}_1$ works on $\mathbf{E}_0$, which is the original set of mean-centred calibration spectra, but from $\mathbf{w}_2$ onwards the weights work on residual spectra. They are not like PC-transform vectors, where each one works on the original mean-centred spectra. In my opinion, if you really want to understand what a particular $\mathbf{w}_i$ is doing then you should consider it in conjunction with residual spectra in the corresponding $\mathbf{E}_{i-1}$.

I didn’t really get there, did I? Personally I think the loadings $\mathbf{p}_i$ are more straightforward to interpret.

1.3.13 The Score Scatter plot



TSG HotCore (Site pm-laptop: Peter Mason) - chat1
File Edit View Window Help
Summary Log Spectrum Stack Scatter PLS
Load Save Cal samples Inputs Run CV
PLS Model (4 factors): Score 1 versus Score 2 (Z-Normalised)
Score 2 (Z-normalised)
1.6
0.8
0
-0.8
-1.6
-1.8
-1.2
-0.6
0
0.6
1.2
1.8
Score 1 (Z-normalised)
Param F
0.973
0.903
0.834
0.764
0.695
0.625
0.556
0.486
0.417
0.348
0.278
0.209

Model this: K_(%) Model: 4 Factors Plot: Score Scatt X: Score 1 Y: Score 2 SDevs: 2 Colour: Param F

Active Samples
1:ch238178
2:ch238179
3:ch238180
4:ch238181
5:ch238182
6:ch238183
7:ch238184
9:ch238186
10:ch238187
11:ch238190
13:ch238192
14:ch238193
15:ch238194
16:ch238195
17:ch238196
18:ch238197

In the algorithm description above, I told you that the scores are “the PLS transform” of the calibration spectra and that you can handle them similarly to PC- or MNF-transform bands. People scatter two PC or MNF bands against each other to investigate extreme pixels, mixing arms, clusters, and so on. You can do much the same thing here with two “PLS transform” scores. The impact of extreme pixels is different, however. In a PC or MNF transform, extreme pixels are sought-after things because they are often the purest pixels in the dataset and the analyst is pleased to find them. In PLS they are unsettling things because they are different to the others. They might be outliers. PLS has a focus on making a robust model and it is unsettling to see any wild calibration samples “out there”, driving the model. A good coverage of calibration samples is what one would like to see.

So here we have a third plot that supports sample outlier removal. The usual mechanisms apply (multiple selection, lasso etc) and the usual caveats apply (*check* possible outliers before discarding).

Select two scores to scatter using the **X** and **Y** lists. The scores to choose from are constrained by the number of factors in your model (Model list).

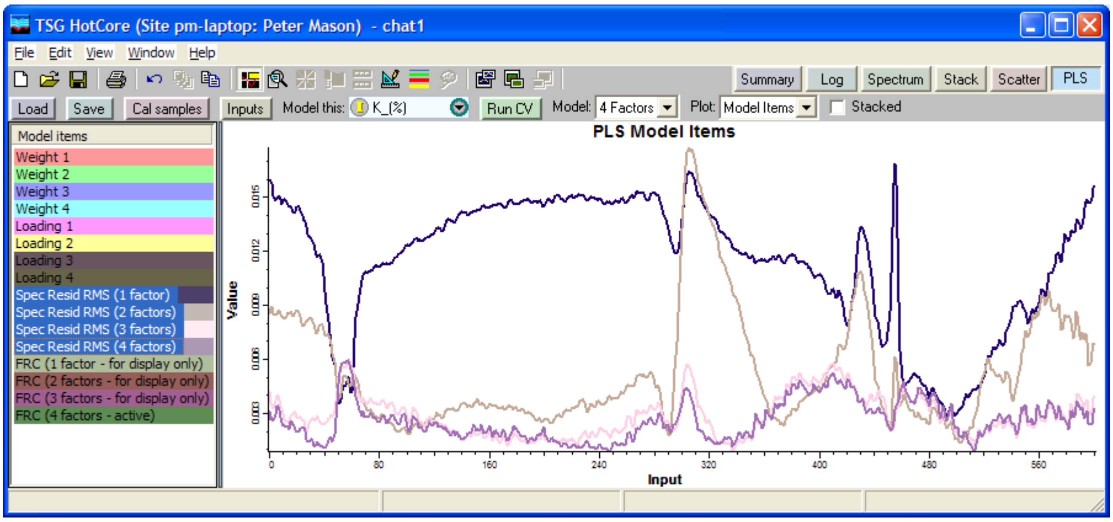
The plot is effectively a Mahalanobis-distance plot because the PLS algorithm, by nature, produces orthogonal scores ($\mathbf{t}_i \bullet \mathbf{t}_j = 0$, $i \neq j$) and each score vector $\mathbf{t}_i$ has been Z-normalised for the plot. (The Z-normalisation gives each score vector a mean of 0 and a standard deviation of 1.) The plot points can be coloured by spectral or parameter residual F (**Colour** list) to assist in outlier detection.

The **green** shading offered here is based on standard deviations, and a threshold is typed into the **SDevs** on-screen control. E.g., With SDevs=2, all points further than 2 standard deviations from the mean (0,0) are in the green-shaded region and they can be selected quickly by right-clicking the list and choosing the “select green-shaded region” item.

1.3.13.1 Data export

If you export the Score Scatter plot to a CSV file or to the clipboard then you get the following for each calibration sample: sample name, param F, spec F, score 1, score 2, ...score m, where m is the number of factors in the final model. (Yes you get all the scores, not just the two selected for the plot.)

1.3.14 The Model Items plot



Model this: K_(%)

Model: 4 Factors

Plot: Model Items

□ Stacked

Model items

- Weight 1
- Weight 2
- Weight 3
- Weight 4
- Loading 1
- Loading 2
- Loading 3
- Loading 4
- Spec Resid RMS (1 factor)
- Spec Resid RMS (2 factors)
- Spec Resid RMS (3 factors)
- Spec Resid RMS (4 factors)
- FRC (1 factor - for display only)
- FRC (2 factors - for display only)
- FRC (3 factors - for display only)
- FRC (4 factors - active)

It’s a collection of things for the current model. It has the weight and spectral loading vectors $\mathbf{w}_i$ and $\mathbf{p}_i$[10]. It also has RMS spectral residual vectors and “final” regression coefficients for each factor level of the model.

You can select one or more of these items for the plot. If two or more are selected then they are normally plotted together to a common scale but if you prefer you can have them stacked by turning on the **Stacked** on-screen control.

[10] i=1..m, where m is the number of factors selected in the Model list

1.3.14.1 Weights

These are the $\mathbf{w}_i$ in the algorithm description above. I've already told you all I want to about them. I'll just repeat that they are forward-transformation vectors to "PLS space", only they work on residual spectra. ($\mathbf{W}_i$ works on the residual spectra left after factor level `i-1`.)

1.3.14.2 Loadings

These are the $\mathbf{p}_i$ in the algorithm description above. I've told you about these too, but I'll repeat or re-phrase some of it because the loadings are interesting.

PLS finds a small number of "pure things" to model combined spectral & concentration variability. The $\mathbf{p}_i$ are the spectra of the "pure things" that PLS has found. But PLS is not a geologist. It doesn't give a hoot about rocks. It doesn't know chlorite from chlorine. Its notion of a "pure spectrum" is purely mathematical. Nonetheless it obviously responds to spectral variability and you will see mineral absorption features in the $\mathbf{p}_i$.

The $\mathbf{p}_i$ are ranked. The first loading $\mathbf{p}_1$ is the most important one and is likely to be the most recognisable spectrum. Sometimes you might get a $\mathbf{p}_1$ that looks like an actual mineral spectrum but more often it'll look a bit deranged. You might see a 1400nm kaolinite feature-set along with an upside-down 1900nm bound water feature, for example. Other early $\mathbf{p}_i$s (e.g., $\mathbf{p}_2$ and $\mathbf{p}_3$) are often good too. Derangement increases at higher factors and before long you'll have $\mathbf{p}_i$s that show little if anything you can recognise. At higher factor levels, PLS will have used up all the good variability and it'll be leveraging peculiar little things for small improvements to its model.

There is no plot of the concentration loadings – the q_is – but you will find them in the plot titles of the spectral loadings.

1.3.14.3 Spec Resid RMS

In the CV pass, one sample is removed from the calibration set, then models are made (one model per factor level) from the remaining samples and used to calculate predictions for the sample that was left out. Each prediction leaves behind a residual spectrum. This whole process is done for each sample in the calibration set, and all of those residual spectra are kept in a safe place. So we end up with a CV (proper)[11] residual spectrum for each calibration sample, for each factor level (nsamples * nfactors residual spectra).

You can look at individual residual spectra in a later plot but here we look at an "average" for each factor level. Any single residual spectrum has positive and negative values so a conventional mean over the calibration set wouldn't make much sense. Instead, TSG gives you the RMS (root mean squared) residual spectrum for each factor level.

An RMS residual spectrum shows spectral variability that PLS left behind, deeming it unimportant to the model. Examining a progression of these residuals together (for factors 1, 2, 3 etc) can be interesting, as it reveals the variability each "next" factor chopped out.

[11] If residual spectra were calculated from the full model that knew every calibration sample then they would be smaller. CV residuals are more useful because they indicate how the model performs when given unknown spectra.

I suppose you could also look at weight and residual pairs (e.g., $\texttt{w}_3$ with $\texttt{resid}_2$) to get a better feel for what the weights are doing.

1.3.14.4 FRC

You've seen this one before, haven't you? It's got a plot of its own – what's it doing here? The one you've seen is the FRC for your chosen model, which has a number of factors given by the Model list. Yes, here it is again but this time it brought some friends – the FRCs for all models at lower factor levels. The point of its reappearance is so that you can see how it evolves as the number of factors increases. Plainly, I want you to select a bunch of FRCs and plot them together. The FRC is an effective item for interpreting what makes a model tick, so examining a progression of FRCs can provide further insight about what each "next" factor does.

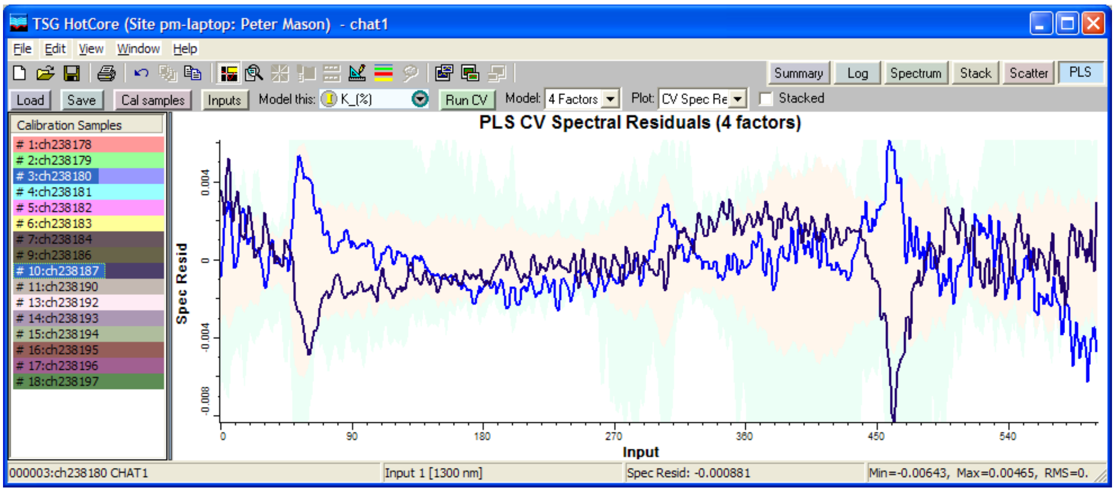
Examining a progression of FRCs might also encourage you to use fewer factors in your final model. Traditional statistics are all very nice but I think it's worth checking things out at a down-to-earth level too. This often happens: The first FRC has an easily understood shape and its coefficients aren't large in magnitude. The second FRC is more extreme (larger range of coefficients) and it's probably busier. And so it goes, with the FRC becoming more extreme and busier as the number of factors is increased. If your "final FRC" has mad ups and downs all over the place and a huge range of coefficient values then never mind PRESS, SEP, R^2 and all the rest of them; I have to wonder if your model will be robust in a production environment. The more extreme the FRC, the more it is balancing things out. You know, "these spectral channels contribute positively and those channels contribute negatively". You don't want a knife edge here. If such a model is given an unknown spectrum that's just a bit different to the calibration spectra then this balancing act could come undone and the prediction could be way off. And as for a busy FRC, the point here, I think, is that analysts don't like black boxes. If an FRC has ups and downs all over the place and you can't understand what it's doing then that isn't reassuring.

Sorry about the rant. So anyway, say TSG says "use the 6-factor model" and you also check out the 5- and 4-factor models, finding that they do reasonably well in the CV Act:Pred plot and their SEP & R^2 values aren't that bad. Come here and check out the FRC progressions. If you see a sharp increase in range and busy-ness (especially range) from factor 4 to 5 or 5 to 6, I suggest you consider using a lower-factor model – the one before the step.

1.3.14.5 Data export

If you export the Model Items plot to a CSV file or to the clipboard then you get one row of values for each input. The first column contains the wavelength for a spectral-channel input or the scalar name for a scalar input. After that you get one column for each item that was selected for plotting.

1.3.15 The CV Spec Resids plot



Model this: K_(%)
Model: 4 Factors
Plot: CV Spec Re
☐ Stacked

000003:ch238180 CHAT1 | Input 1 [1300 nm] | Spec Resid: -0.000881 | Min=-0.00643, Max=0.00465, RMS=0.

This plot shows you individual residual spectra for the calibration samples, at the currently-selected factor level (Model list).

See the "Spec Resid RMS" section above for a reminder about what these things are and why I call them "CV spec resids" instead of plain "spec resids".

You can select one or more for the plot. If two or more are selected then they are normally plotted together to a common scale but if you prefer you can have them stacked by turning on the **Stacked** on-screen control.

In an overlay plot (Stacked turned off) you get some shading in the background. The green-shaded region is the overall spectral residual envelope – the highest and lowest residual values found for each input in the whole calibration set. The red-shaded region is the spectral residual RMS[12] for the calibration set, mirrored for negative values.

If you left-click on the plot then TSG will report on the sample closest to where you clicked. If you have a floater going then it will navigate to this sample.

If you are curious about spectral residual outliers then personally I think this spectral view is much richer than the single Spec Resid F number you've seen before in other plots.

1.3.15.1 Data export

If you export this plot to a CSV file or to the clipboard then you get one row of values for each input. The first column contains the wavelength for a spectral-channel input or the scalar name for a scalar input. After that you get one column for each sample that was selected for plotting.

[12] Root mean squared. It has values >= 0.

1.4 Prediction – the 'PLS' scalar

TSG's scalar construction wizard has a new method – "`PLS: a PLS prediction result calculated using a PLS calibration file`". It predicts some scalar (e.g., potassium %) from the dataset's spectra (or other scalars) using a PLS model that was made earlier. It is similar to the `STAT` and `AUXMATCH` methods in that it requires a special external file (in this case a PLS session file) and it makes a scalar that you can recalculate ("modify scalar" menus) and take across in Copy Processing. Before I describe the scalar I shall tell you some things about...

1.4.1 The PLS prediction algorithms

PLS *calibration* comes up with a model, given a calibration set of inputs (normally just spectra) and corresponding concentration values (the scalar that was modelled). See earlier for an algorithm description. Here's a relevant summary. PLS calibration factorises input matrix $\mathbf{X}$ and concentration array $\mathbf{y}$, finding loadings $\mathbf{p_i}$ & q_i, scores $\mathbf{t_i}$, and leaving residuals $\mathbf{E_i}$ & $\mathbf{f_i}$:

$$\mathbf{X} = \mathbf{t_1 p_1}' + \mathbf{t_2 p_2}' + \ldots + \mathbf{t_m p_m}' + \mathbf{E_m}$$
$$\mathbf{y} = \mathbf{t_1} q_1 + \mathbf{t_2} q_2 + \ldots + \mathbf{t_m} q_m + \mathbf{f_m}$$

M is the number of factors in the final model. Each $\mathbf{p_i}$ is a vector with the same dimensions as an item in $\mathbf{X}$ (i.e., it's like a spectrum), and each q_i is a scalar. The scores $\mathbf{t_i}$ are common to *both* the $\mathbf{X}$ and $\mathbf{y}$ factorisations, although it is useful to think of them as "the forward PLS transform" of the calibration inputs $\mathbf{X}$. A score vector $\mathbf{t_i}$ has one element per calibration sample. Another set of items found, not shown in the factorisation above, is a set of weight vectors $\mathbf{w_i}$. $\mathbf{W_i}$ is "the forward transform vector" for factor level i and it works on the input residuals $\mathbf{E_{i-1}}$. $\mathbf{W_i}$ has the same dimensions as $\mathbf{p_i}$ (i.e., it's also like a spectrum).

It's easiest to understand how prediction works by looking at...

1.4.1.1 The slow prediction algorithm

We come in with an unknown input $\mathbf{z}$. $\mathbf{Z}$ is just like a sample in the calibration matrix $\mathbf{X}$, only it is not an actual sample from $\mathbf{X}$ – it was probably measured on a different occasion. If PLS calibration was done on spectra (no input scalars) then $\mathbf{z}$ is just a spectrum. If it included input scalars then $\mathbf{z}$ includes compatible scalar values. If any spectral subsetting or processing was done then $\mathbf{z}$ has had the exact same treatment.

The idea behind PLS prediction is straightforward:

1. Subtract the calibration's mean input vector from $\mathbf{z}$. (We did this to each sample during calibration.)
2. Transform $\mathbf{z}$ to the calibration's "PLS space" to get scores s_i, $i=1..m$. (Each s_i is a scalar.)
3. Use the y factorisation equation to predict a concentration. That is, $y=\sum_i s_i * q_i$, $i=1..m$. (The q_i come from the calibration and we just found the s_i.)
4. Add the calibration's mean concentration to this prediction. (We subtracted it during calibration.)

Unfortunately, we don't have "forward PLS transform" vectors that work on $\mathbf{z}$ in a straightforward way, like one does when doing a PC transform. We have transform vectors $\mathbf{w_i}$ where each one works on the previous ($i-1$) factor level's spectral *residual.*

Therefore, steps 2 & 3 above are done in a loop over the factors ($i=1..m$), using bits of the calibration algorithm. To start off, set $\mathbf{z}_0=\mathbf{z}$ and $y=0$. Then:

- $s_i = \mathbf{z}_{i-1}' \# \mathbf{w}_i$
- $y = y + s_i * q_i$
- $\mathbf{z}_i = \mathbf{z}_{i-1} - s_i * \mathbf{p}_i$

...where $\mathbf{w}_i$, $\mathbf{p}_i$ and q_i are items from the PLS calibration.

And that's it – the slow PLS prediction. A useful side effect is that we are left with the unknown's spectral residual $\mathbf{z}_m$ at the end. It can be used for quality control.

1.4.1.2 The fast prediction algorithm

It can be shown that a "final regression coefficient" vector **frc** can be distilled out of the calibration items $\mathbf{w}_i$, $\mathbf{p}_i$ and q_i, $i=1..m$. I won't show it because I'd be lying if I did – I can't keep the proof in my head for any decent length of time. Instead I refer you to Helland's paper[13] again.

Anyway it turns out that:

$$\mathbf{frc} = \mathbf{W} \# (\mathbf{P}' \# \mathbf{W})^{-1} \# \mathbf{q}$$

Where $\mathbf{W}$ is the matrix made out of the weight vectors $\mathbf{w}_1..\mathbf{w}_m$ stuck side by side, $\mathbf{P}$ is the matrix of spectral loading vectors $\mathbf{p}_1..\mathbf{p}_m$, and $\mathbf{q}$ is the vector of concentration loadings $(q_1..q_m)'$.

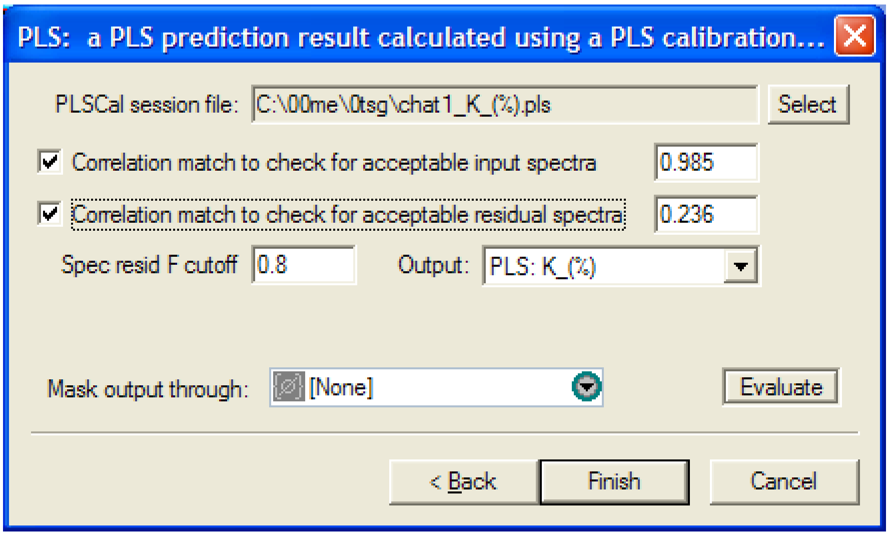
Prediction is done with **frc** as follows:

1. Subtract the calibration's mean input vector from $\mathbf{z}$. (We did this to each sample during calibration.)
2. $y = \mathbf{frc}' \# \mathbf{z}$. (This is just a dot product between **frc** and **z**.)
3. Add the calibration's mean concentration to this prediction. (We subtracted it during calibration.)

A drawback with this method is that it just predicts a concentration – it does not leave behind a spectral residual for **z**.

1.4.2 Making a PLS prediction scalar

Open the scalar construction wizard, select the PLS method, and click "next".



PLS: a PLS prediction result calculated using a PLS calibration...

PLSCal session file: C:\00me\0tsg\chat1_K_(%).pls

☑ Correlation match to check for acceptable input spectra 0.985

☑ Correlation match to check for acceptable residual spectra 0.236

Spec resid F cutoff 0.8

Output: PLS: K_(%)

Mask output through: [None]

Well you won't see the whole thing straight away. Most of the controls will remain invisible until you have selected a compatible PLSCal session file. Once you've

[13] I.S. Helland, Commun. Statist. -Simula, 17(2), 1988, pp581-607: "On the structure of Partial Least Squares Regression".

done that you can set up some quality control filters if you like (there are three of them), evaluate sample plots in the floater, select an output (there are four output options) and generate the scalar. As is normal for a calculable scalar in TSG, you can put the PLS scalar through an output mask if you like.

1.4.2.1 PLSCal session file

This is something you made earlier, probably from a different dataset. You made it during PLS calibration by clicking the Save button in the PLS screen. All sorts of things are remembered in this file. The ones relevant to PLS prediction are:

- All the things set up in PLS calibration's "inputs" dialog: The spectral channel wavelengths or resampling specs and the spectral processing steps (if spectra were selected); the names and types of the input scalars (if input scalars were selected); the "input standardisation" setting.
- The calibration's **w**, **p** & q items[14], the average calibration input, and the average calibration concentration. (These are used by the prediction calculation.)
- All of the preprocessed calibration inputs. (These are used for the first quality control filter.)
- All of the calibration CV spectral residuals. (These are used for the second quality control filter.)
- The name of the concentration that was modelled. (This provides a default name for the prediction scalar you're making now.)
- Some stats.

1.4.2.1.1 Compatibility

You can't use a PLSCal session file for prediction on just any other TSG dataset. E.g., you can't use a VSWIR session file on a thermal dataset.

- If the PLS calibration used spectra then the prediction dataset must include the wavelength coverage (min to max) that was taken in calibration. However, it does not need to have the same channel spacing as the calibration dataset; the spectra will be resampled to be like the calibration spectra if necessary. Also, it does not need to have the same wavelength units. Wavelength unit conversion will be done if necessary, and if the converted units give the required coverage then the spectra will be accepted.
  That said, for best results the prediction spectra should be just like the calibration spectra, ideally measured by the same spectrometer.
- If the PLS calibration used any input scalars then the prediction dataset must include numeric scalars with the same names.

If spectra are used then PLS calibration always takes the first spectral layer, which is usually "reflectance", and does its own processing (hull quotients or whatever). The same process is followed by PLS prediction – the same processing steps (as recorded in the session file) are applied to the prediction spectra. So if the two datasets have compatible "reflectance" spectra then their processed spectra will also be compatible. If the two datasets were measured by different spectrometers then their reflectance compatibility might be in question. A difference in signal-to-noise ratio is an obvious issue. A more sinister one is a difference in what "reflectance" means; "absolute reflectance" versus "reflectance relative to a particular piece of spectralon",

[14] Not all session files contain these items, because you can save a PLS session before running CV & generating model data. Such a session file won't be accepted for PLS prediction.

for example, or "bidirectional reflectance" versus "hemispherical reflectance". There can also be a difference in response linearity. These differences can be subtle but they might have an adverse affect on PLS prediction.

If you succeed in selecting a compatible PLSCal session file then the other controls on the page will become visible.

1.4.2.2 Quality-control options

If you read the first part of this document, you should have picked up on some real concerns about PLS prediction accuracy. In PLS calibration you could get a good idea of this because you actually *had* the values that you were trying to predict, but now you don't, obviously. (If you did then why are you here?) Bad PLS predictions can be damaging in a production environment. Here are a few bullets travelling through time, seeking a target in the future. Or maybe they're clones from Excel Saga. Either way, they're gonna get you.

- That man said: "Yo!" Some others said: "Here we are!" He said: "The PLS calibration set must be *representative*." They said: "We understand." He continued: "It must include *all* spectrally-active components you might find when dealing with unknowns later on (in PLS prediction)." They said: "Of course!", "Right you are!" and "I'll see to it!" He concluded: "This is *crucial!*" They said: "No worries mate!" In their hearts, however, they laughed him off, feeling that his demands were unreasonable and probably of the soft-handed academic variety.
- That man over there said a bunch of stuff about monitoring spectrometer performance, taking good care of transfer standards, observing operating procedures intelligently, and so on. They laughed him off too, for similar reasons.
- That other man said: "No. Do not measure those chips without the chip-tray mask or the chip-tray plastic will contaminate your spectra and you'll be sorry later." And... you know.

So it goes that, for one reason or another, I happen to know you will get some spectra in PLS prediction that are unlike any used in the PLS calibration. And that's a bad thing. Predictions made from such spectra will probably be wrong, but you wouldn't know how wrong. (Otherwise, again, why are you here?)

The best Q/C that PLS prediction can offer is some sort of check for spectral suitability. The traditional measure is spectral residual F and TSG offers two other correlation-based ones.

1.4.2.2.1 Spectral residual F

This is the same measure as the "spec resid F" you've seen before in the calibration's "CV Act:Pred" and "CV ParF:SpecF" plots.

PLS prediction leaves behind the unknown's residual spectrum and the sum-of-squares across inputs (spectral channels) is calculated from it. This is compared with the calibration set's average spectral residual sum-of-squares using an F probability. The F probability says:

- This unknown's spectral residual is probably bigger than the cal average.

Or...

- This unknown's spectral residual is probably too big.

Or...

- This unknown is probably strange. You should probably ignore its prediction.

I am reluctant to give you a cutoff to use just like that. The default is the highest CV spec resid F found in the calibration set itself and you shouldn't count on it being suitable for prediction. I know that sounds weak but spec resid F just isn't a direct QC measure for prediction, so you simply have to find a good cutoff yourself by experimentation, if you can. (For starters I suggest a high cutoff, e.g., `0.9` or more, so that you at least get *some* results.) Once you have decided on a cutoff, type it into the **Spec resid F cutoff** field. Prediction samples that get a higher spec resid F than this cutoff will get a NULL result.

Note that this check is always done. To make it do nothing, type in a cutoff of `1`.

While discussing PLS calibration I said that spec resid F is a poor second cousin to param resid F. Obviously you don't have param resid F here so you'll just have to make do with what you've got. It's a gross measure. It doesn't check *where* (e.g., which spectral region) the residual is too big; it just checks if the *overall* residual is too big. It's better than nothing, though.

1.4.2.2.2 Correlation match on input spectra

If the prediction's spectrum is supposed to be like the ones used in calibration then why not check this directly? That's what we have here. A correlation[15] match is done between the prediction spectrum and each calibration spectrum. The best (highest) correlation is found and compared against a threshold. If it's not good (high) enough then the prediction returns a NULL result, because the prediction spectrum is unlike any used in calibration.

This check is a bit simple-minded because it does not allow for some linear mixing that the PLS model might actually accommodate.

To activate it, turn on the **Correlation match to check for acceptable input spectra** checkbox and type a threshold into the adjacent field. The higher the fussier, with a threshold of `1` meaning that the prediction spectrum must match a calibration spectrum exactly (which is unreasonable). The default threshold is dredged from the calibration set somehow and isn't that good. Setting this threshold correctly will require some exploration. This is discussed further under Output below.

1.4.2.2.3 Correlation match on residual spectra

This is similar to the above but it works on residual spectra, not full spectra. The prediction algorithm takes an unknown spectrum, does its thing, and leaves behind a residual spectrum. A correlation match is done between this residual spectrum and each calibration residual spectrum (kept in the session file). The highest correlation is found and compared against a threshold.

In other words, this is a *spectral* match done on the bits of "junk" left behind after PLS has removed the spectral variability it wants. It is hoped that it might complement the spec resid F test with some spectral intelligence – i.e., is the "junk" in the *same place* and with the *same shape*, not just how much is there altogether.

To activate it, turn on the **Correlation match to check for acceptable residual spectra** checkbox and type a threshold into the adjacent field. Again the default threshold isn't that good and a viable threshold will require some exploration.

[15] Pearson's correlation coefficient, like the default method used in TSG's "Auxmatch" functionality.

1.4.2.3 Output

There are four options in this list. The first three are “input correl”, “spec resid correl” and “spec resid F”. They return the measures used in the three quality control tests that can be done. The last option, the default, is named after the scalar that was modelled during calibration. It returns the actual prediction result and it’s what you came for.

1.4.2.3.1 Finding Q/C thresholds

Before you release a PLS model for running predictions in a production environment, it is in everyone’s best interests that you, the model’s author, establish Q/C checks so that dud predictions are avoided as much as possible. To do this, you will have to find effective thresholds for the checks. You’ll need a test dataset containing samples that you are able to judge. I can’t advise on this except for the obvious – if these samples have known concentration values but you didn’t use them in the calibration then that’s ideal.

The first three output options are there for you to explore thresholds for the QC checks you have chosen to do. (Well the spec resid F check gets done whether you like it or not.) The idea is to run the PLS scalar construction more than once, making more than one scalar – one for the actual prediction and one for each test you want. Then simply go and look at all these scalars and see if you can find Q/C thresholds that isolate dud predictions. If you do this in TSG’s Log screen then maybe you can make a User-class Mask scalar, bring it up for interactive editing, and use the “selection by scalar match” tool to help visualise what your thresholds are doing.

The end