

Introducing INCA Discovery™: A Sophisticated Software Package for Constrained Linear Model Identification

Background

Announcement: IPCOS has launched its replacement software for INCA Modeler. The new product is called INCA Discovery™. The objective of the development project was to build a model identification package that can discover good clean models in a step test data set by automating most of the workflow steps involved.

Introduction: Accurate dynamic process models are used in countless simulation and control applications. For example, in order to build a good multivariable controller we need to find accurate models between the Manipulated Variables (MVs) that the controller manipulates, and the Controller Variables (CVs) that the controller has to keep between user defined limits. In addition, we also need to take Feed Forward signals (FFs) into account during the model identification process in order to predict the effect of external disturbances. [Naming Convention: We group manipulated variables (MVs) and feed forward variables (FFs) together, because they are both “model inputs” and we call them “Independent Variables”. As a result, the CVs are also referred to as “Dependent Variables” and constitute the model outputs.]

Multivariable PID Loop Tuning: For process applications where multiple PID loops interact, the best approach is to use a multi-variable loop tuning purposes, like the AptiTune™ software package. This model based approach requires multivariable (MIMO) models between the PID outputs (or OPs) and the PID Process Variables (PVs). [MIMO refers to multiple-input-multiple-output models, or “multivariable” for short.]

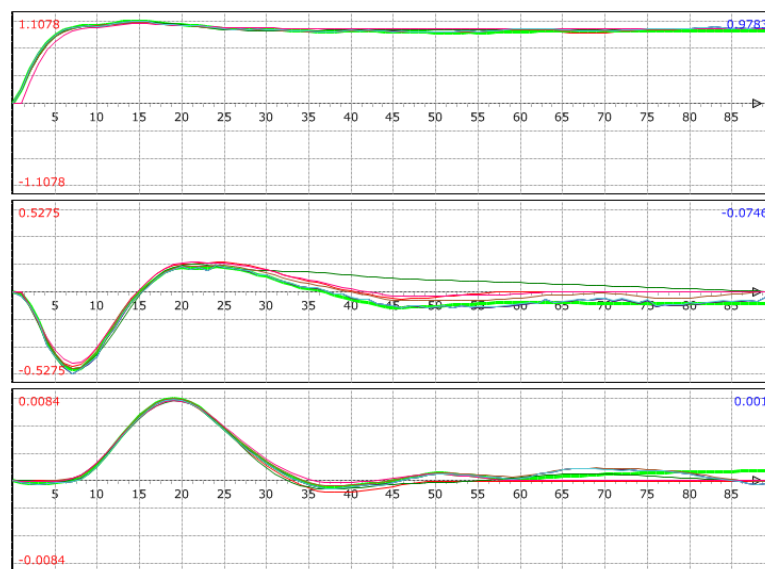
The Need for Process Realism: The perfect model identification package will be able to fit smooth realistic process models between the inputs and outputs, in such a way that every “squiggle” in the model is real and is a close reflection of the actual process dynamics (perhaps due to PID interactions or heat integration), even in the presence of large disturbances and significant measurement noise. This is quite a challenge and can be very difficult to achieve in practice, so it is important to allow the user the ability to impose his or her process knowledge on the model identification problem. Prior knowledge may include knowing the following information for some or all of the model curves:

- **Dead Time:** How long does it take before a change in an upstream MV or CV *reaches* the backend of the process and start to affect the product quality in the last column?

- **Process Gain:** We often know the sign and the approximate range of the process gain value, either from information provided by a rigorous model, or from the controller structure. For example, the gain between the SP and PV of a closed PID control loop is unity (1.0).
- **Time to Steady State (TTSS):** Some process relationships have a short TTSS value because there is not much holdup in the process, while other relationships are very slow due to large vessels and thermal inertia in the system. A “one-size-fits-all” approach does not work well.
- **Null Models:** Often we know that there is no model curve present for a specific MV/CV relationship.

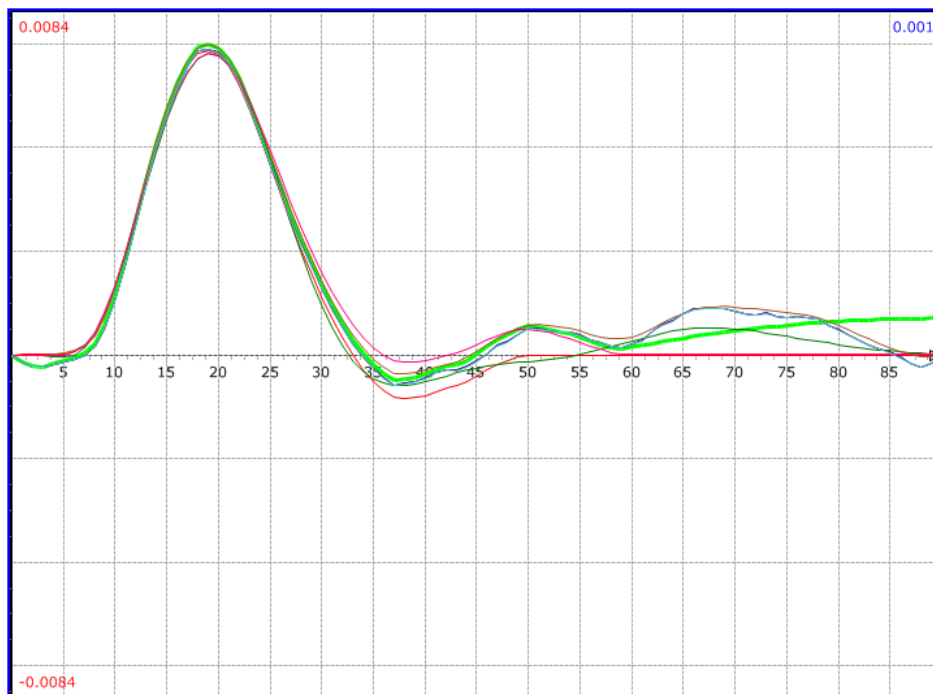
What is missing from today’s crop of model ID packages? The first Windows based model identification (ID) packages appeared around 1996. Since then, a number of innovations have been introduced with varying degrees of success. The following is a list of potential shortcomings in current model ID packages:

1. **Process Gain Specification:** We know that the gain between the SP and PV of the same PID loop is unity (1.0), while all other inputs will have zero gain. For level loops, we also know that the OP models are zero gain everywhere. Occasionally, we can see from the data that the identified gain is too small (perhaps due to non-linearity) and then we want the ability to increase the gain value manually. The example below is a model of a level controller measurement (PV) to changes in Setpoint and 2 disturbance variables (two upstream level controller setpoints). Before we start, we already know what the gain should be for each response



Note that the first model represents the SP to PV model. The identified gain is 0.978 (should be 1.000). The remaining models should all have zero gain.

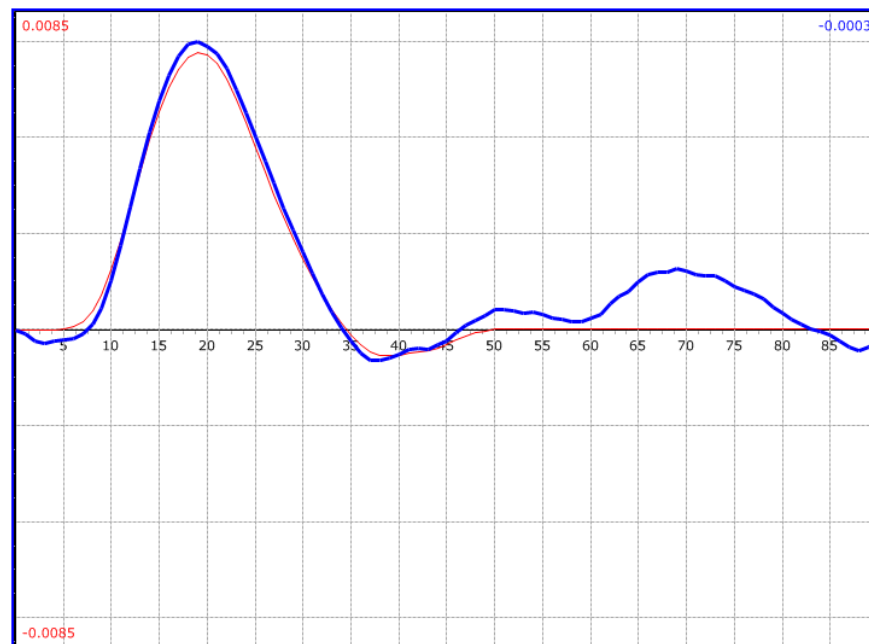
- Dead Time Specification:** We often know the approximate dead time of a specific MV/CV relationship. For example, for product quality measurements, we know there is a measurement delay as the material with the new composition travels through a set of reactors and distillation columns, through the analyzer sampling line and finally the Gas Chromatograph which will take a predetermined amount of time to analyze the sample. From a small number of steps, we can often estimate the dead time by inspecting the data. Most model identification packages do not allow the user to specify the dead time, and the identification algorithm will often find a slight improvement in its cost function by fitting a slight inverse response, which means the gain estimate is also wrong (the entire curve moves up or down). We would get a better result if we could tell the model ID engine what the actual dead time is. Below, note that the model curve has 7 minutes of pure dead time, and Time-To-Steady-State of about 60 minutes.



- Null Models:** We often know from the physics of the process that a certain MV/CV relationship will be zero. If we are not able to tell the model ID engine that there is a “Null” model, it will always try its best and create some phantom model curve, often small and very noisy (and complete wrong – of course!).

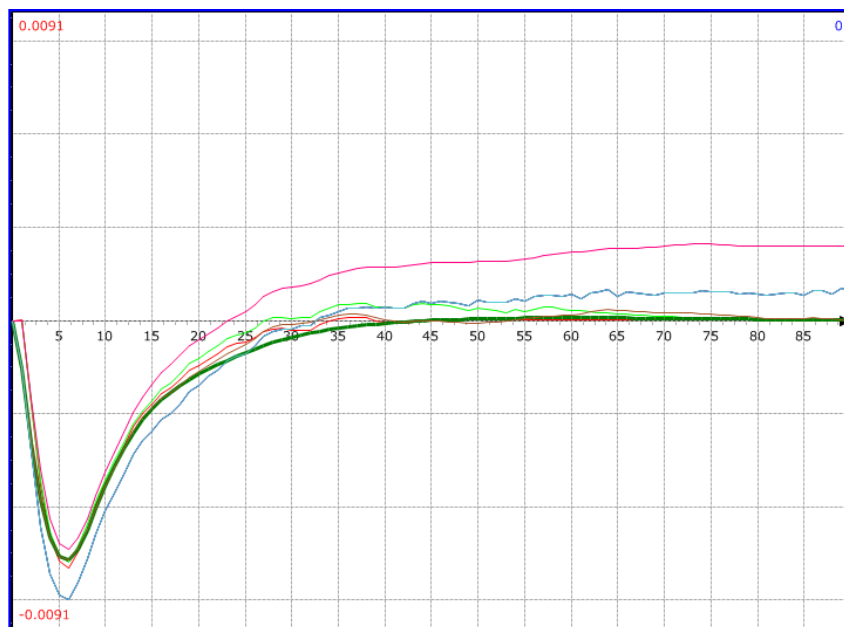
4. **Specification of Time to Steady State:** Most model identification packages allow the user to set Time to Steady State (TTSS) on a per trial basis. However, all process units have a combination of both slow and fast responses, so we can do better if we specify the most appropriate TTSS on a per-curve basis.

5. **Prevent Inverse Responses:** We often know from the physics of the process that an inverse response is not possible, yet we see these phantom inverse responses where the CV first goes in the wrong direction before turning around and lining out at the gain value. These false inverse responses are often caused by Feedback Correlation where some MV moves during the step test were made in response to unknown disturbances, in order to keep the CVs between safe limits. As a result, the MV becomes correlated with the disturbance. Since the disturbance signal is NOT available for inclusion in the case as an additional Independent, the model ID algorithm can “improve” the prediction slightly by “cheating” i.e. bending the model curve in the opposite direction to partially explain the effect of the unknown disturbance. The net result is that the curve is offset due to the presence of the inverse dynamic, and the gain value will also be too low. Gain errors are small as 10% (which are common) can dramatically change the RGA values of the final model matrix, leading to major problems for the steady state LP or QP solver. For this reason, some (but not all) model gains must be as accurate as possible.



Note the inverse response in the front part of the model (which should have been dead time). The model shown in red is closer to reality and was found by imposing dead time, TTSS and gain constraints.

6. **Ensure Monotonicity:** Most model “curves” have fairly complex dynamics in the front part of the model with several direction changes, but after some point, the response will not change direction anymore (becoming monotonic) and slowly ramp up or ramp down to the final gain value. We can often “see” the true model hidden by noise and ugly squiggles in the identified FIR model curve, but in the past it was hard for the user to impose this insight on the model ID problem.



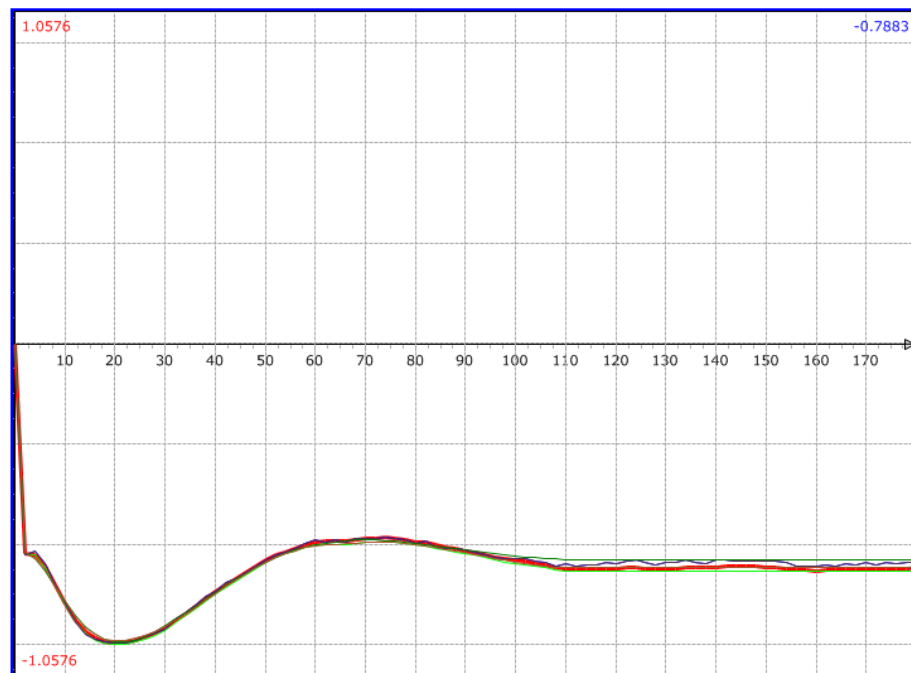
Note that the model curve shown in green (produced via the Adaptive FIR method) is monotonically increasing from time value=6 minutes until it finally lines out at a zero gain. The remaining models are not monotonic and as a result less accurate.

7. **Limit Model Complexity:** The user normally has a good “feel” for how complex a model curve should be. In the standard FIR approach, the identified model curve is often a lot more complex (more noisy and squiggly) than we know it should be. The ideal model ID package will allow the user to set the degree of complexity of the model. This can be done by specifying model order on a per curve basis.
8. **Mix & Match Capability:** Many users remark that if you try out two alternative model ID technologies, you often find that some model curves are better using one method, while the best model curves for

the rest will come from the alternate technique. The question then arises as to whether we can mix and match different model ID algorithms? The current crop of model ID algorithms do not allow for this.

9. **Dealing with Short Periods of Bad Data:** Every project usually has several periods of bad data of a few minutes where the process was responding normally, but the instrument or the data acquisition system failed for a few samples. In the standard approach, FIR models with long TTSS takes a LOT of data to re-initialise after a bad data slice. This can be a major disadvantage of the FIR approach. In addition, most model ID packages do not distinguish between a real process disturbance (like soot blowing on a furnace) where the test data is obviously of no use (i.e. poisonous!), and instrument problems where either the instrument provided bad or no readings for a while, or the data collection system simply missed a few samples. If 5 minutes of instrument problems are marked as “bad”, and TTSS is 3 hours, then a total of 3 hours and 5 minutes of data is usually lost. This can be very expensive as 7 or 8 incidents like this per day can make the entire step test for the day useless. In this case, additional step test data needs to be collected to compensate for the loss of data.
10. **Remove Independents that Do Not Contribute Enough:** We often include feed forward signals in case they will help to clean up the models. However, a large number of weak feed forward signals (especially if they are correlated) can actually hurt the FIR model more than it will help it, as the parameter count goes way up and the condition number of the Least Squares problem can deteriorate a lot - or even become singular leading to an error. Ideally, the model ID package should tell us which MVs or FFs are not worth keeping, and these should be removed automatically.
11. **Partial Predictions:** In order to validate the model, the user will often want to see the effect of one MV only on the predictions, or may want to remove a few weak MVs and FFs to see if that degrades the prediction in a significant way, or not. In this way, we can clearly see if the particular Independent is contributing enough to keep it, or if we should remove it.
12. **Identify Direct Kicks even when using Over-Sampling:** If the Time to Steady State is fairly long (e.g. more than 3 hours), we are normally forced to over-sample the data (change from 60 second sampling to say 3 minute sampling). However, if some (or even just one) of the models have a large initial kick, or a sharp inverse response, or a fast under-damped response, then slow over-sampling will introduce large errors into the front part of the models. Most model ID packages struggle with a combination of fast and slow dynamics. The ideal

model ID package will use high resolution for the front part of the model, and progressively lower and lower resolution for the backend of the model curve.



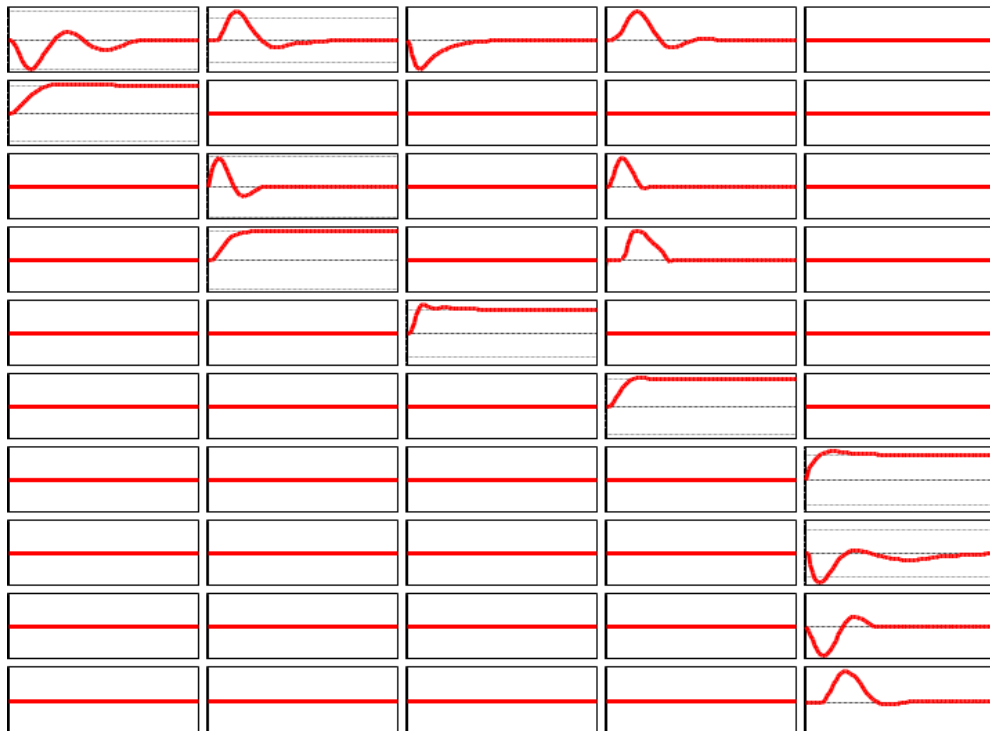
Both smoothed and unsmoothed FIR models are shown. Note that the direct kick in the PID OP has been captured correctly, while the rest of the model curve is realistic and smooth.

13. **Suppression of Drift Disturbances:** Many process applications are plagued by slow drift disturbances for which we do not have a Feed Forward signal. An example is slow feed quality changes due to stratification in the feed tank. Ideally, we would like the ability to suppress these disturbances and reduce the impact of the unknown disturbance on the model ID results.
14. **Unequally Spaced FIR Coefficients:** One problem with FIR is the large number of parameters. More FIR (impulse) response coefficients are needed for the front part of the model curve. The back-end part of the curve is normally very smooth, needing far fewer coefficients. Ideally, we want variable parameter spacing so we can have more coefficients in the front part of the model curve, and progressively wider spacing toward the end.

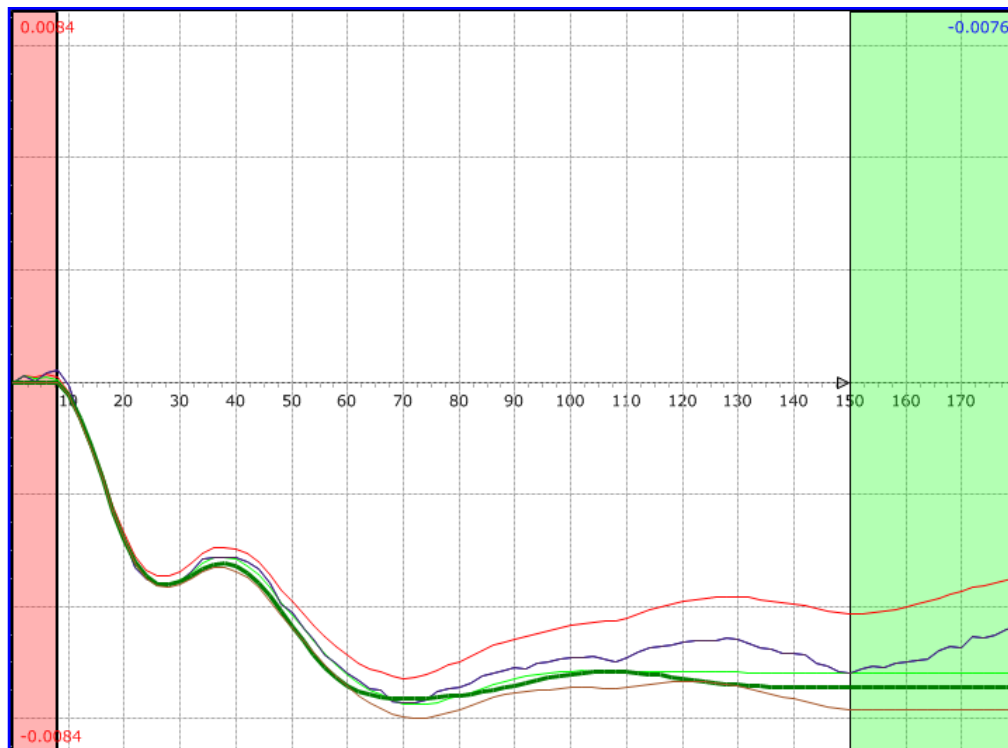
Summary of New Features in INCA Discovery

Here is a summary of the new features we support for version 1.0:

1. **Scalability:** The program was designed to work with thousands of data “signals” (DCS tags), and data sets spanning several months or several years, so most objects are kept on disk. The largest project (so far!) has 14000 signal, with several months of one minute data. Objects are kept on disk. Temporary copies are kept until you click “Save” (so that is your undo button). Even if you get a power failure during a save, the files should not be corrupted.
2. **Usability:** We can drag individual signals or entire signal lists into the Independents/ Dependents tab to populate a case. This is much easier than searching for a signal at a time. We can also search for tags by using the * as wild card (e.g. 38TC00* or 38TC*.PV). “Type-ahead” is also supported to search for tags. If you double click a signal or a case, you will see a plot.
3. **Final Model Matrix:** We can drag and drop model curves into our final model matrix and then export it in several formats. This allows us to mix and match model curves from different cases and different trials.



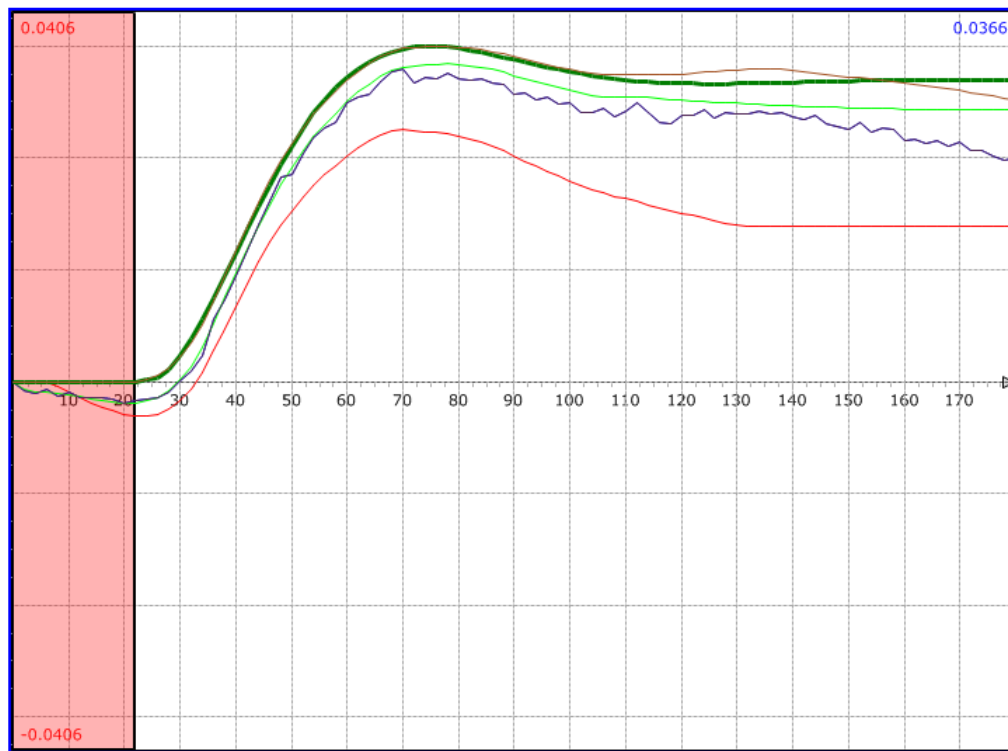
4. **Curve Constraints:** For all the model ID algorithms, we can set TTSS, minimum dead time, and model gain constraints (on a per curve basis), for all or some of the model curves.
5. **Parametric Transfer Function Models:** We have added a Hankel matrix reduction (FIR to State Space model conversion) with automatic model order determination, so noisy FIR curves can be converted into a series low order transfer functions which will be a lot smoother. The typical model order range is normally between 1 and 5, but the user can pick a different range.



Note that by imposing a dead time specification of 9 minutes and a TTSS spec of 150 minutes, we managed to fit an accurate 5th order model that accurately approximates the original FIR model. In this case, the Transfer Function is much more realistic than the noisy FIR result.

6. **Automatic Dead Time Detection:** The Automatic FIR method uses quadratic polynomials to fit a smooth FIR model with improved process realism. It contains automatic dead time detection that work quite well. The user can still edit the minimum dead time if required. We can take control of dead time and line spacing (so we can selectively turn off the automation) by setting the dead time specs where we want them, and

then using polynomial spacing factor like 6 or 12 to force progressively wider line spacing. In this case, less quadratic curves will be used in total.

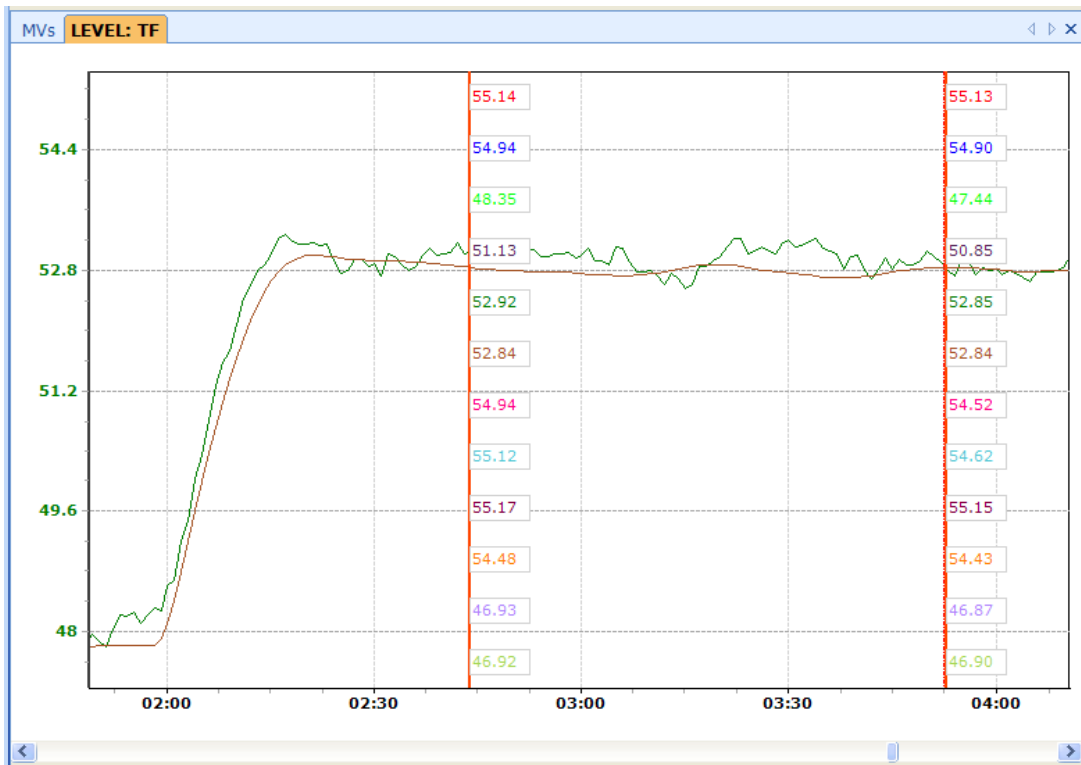


Note that the dead time detection worked well. If the user wants to edit the dead time estimate, then that is possible too.

7. **Optimization of Execution Speed:** All the available processor cores will normally be used since we use the famous INTEL machine code library for all matrix calculations.
8. **Locked Curves:** Weak or non-existent model curves can be defined as “Null Curves” via a right click option. Also, we can lock any model curve (handled as an equality constraint in the QP solver) and we can then edit the FIR smoothing factor, and run it again. We will move this number to a more prominent place.
9. **Copy and Paste a Model Curve into another Trial:** We can copy and paste curves from one trial to another and then lock them (as long as they have the same TTSS and sampling period). This allows us to move curves from a Hankel FIR2SS model reduction trial into the FIR or Adaptive FIR trial, so we can now mix and match different methods

to get the best of both the parametric (transfer function) world and the non-parametric (FIR) world.

10. **Slicing is Done in a More Efficient Way:** For FIR/Automatic FIR, if the bad data slices are all the same, then one Hessian is built and solved as a series of single MISO problems that share the same Hessian. If the slicing is different, then the trial will be broken up and solved as a series of MISO models, and each CV will have its own individual slicing, so less loss of data. The algorithm will re-initialize after an MV slice, but CV slices are simply removed from the Hessian and no data is lost due to re-initialization.
11. **De-Trending to Remove Drift Disturbances:** This is optional for FIR, Adaptive FIR and Transfer Function approach as we differentiate the data, but we can use both pre-processing options to further suppress drift disturbances in the FIR and Transfer Function Method.
12. **Calculations are Supported:** The list of formulas is provided to create calculated signals. It has an Excel-like look and feel.
13. **More Flexible User Interface:** We can dock and undock time trends and rearrange the windows in many different ways. This is very useful if you have two monitors side by side. Trend can go away and be brought back as needed. We can also rearrange the desktop any which way we like by dragging a window. We tend to do slicing on the prediction plots, as that makes it easier.
14. **Model Formats:** INCA Discovery can export models in most of the popular formats.
15. **Signal Formats:** Discovery can import .vec, .clc, and Excel .csv files as well as INCA Test files. We do not require that all signals have the same start or end time. We can merge signals and add new data at the start or the end of the signal. We can even mix and match signals in the same case that have different sampling period, different start and end dates, and the code will interpolate for you so you can pick any sampling period (and it is not necessary for the new sampling period to be a multiple of the original sampling period).



Discussion and Demonstration of New Capabilities

Process Realism: We have used a statistical approach based on model uncertainty to automate dead time + TTSS detection and to determine if strange artefacts and cycles late in the model curve are real or not. If they are unlikely to be real, the code will use a series of widely spaced quadratic polynomials to approximate the model curve, with the spacing of the polynomials dependent on the model uncertainty bound around the nominal model curve. In the past, a single bad disturbance that was not sliced out could have introduced an undesirable squiggle towards the backend of the model curve. This is now taken care of, our models are less sensitive to bad data than before.

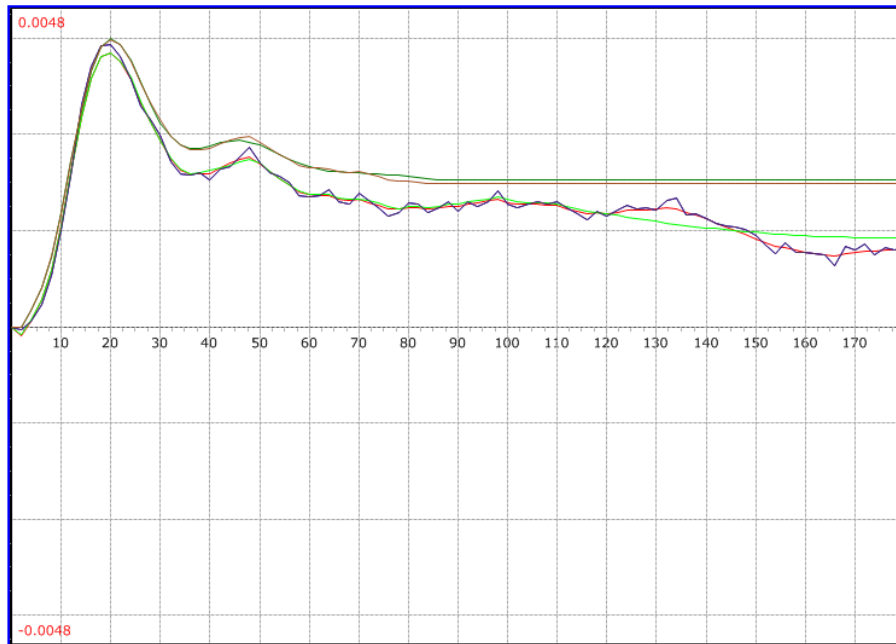
No Model Present: We can also selectively mark very weak (unusable) responses as "Null", meaning they will be removed from the model ID problem. As a result, we can combine many CVs in one case and selectively mark which MVs affect each CV. We also use individual slicing per CV rather than the usual common slicing. This means we do not need to set up one small case per CV in order to make use of all the available data.

Distinguish between MV slices and CV slices: An MV slice means the process was not responding normally (e.g. during soot blowing on a furnace) and the data is unusable. In that case, we will re-initialise after the bad slice (and we lose some data). If we introduce a CV slice, we assume the process was essentially OK but that the instrument was reading incorrectly and we remove the affected data points. We do NOT have to re-initialise after a bad slice so we lose a lot less data that way. This is a lot more efficient than the traditional common slicing approach.

Use a Quadratic Program (QP) to Impose Constraints: We use a QP to solve the model ID problem, so we have the ability to impose many kinds of equality and inequality constraints. You can lock certain model curves so they do not get re-identified (useful during a retest when you only want to move a subset of the MVs and re-identify those models). You can even mix and match the different model ID algorithms: You can copy a Transfer Function model curve into the FIR method, lock it, and then rerun the FIR trial, possibly with different smoothing. We can copy a model curve from an imported model and then paste it into our new case, lock it, then rerun the case. These locked FIR coefficients are then treated as equality constraints. These features are very useful when one model ID algorithm gets the better models for some relationships, but another does better on the remainder. For a revamp where you only want to move some MVs, but due to disturbances you are forced to move other MVs too, you can use the old models from the imported model, and only identify the new model curves you are looking for. This makes a revamp project much easier than before.

Model Identification Technology: We support a total of three algorithms at present:

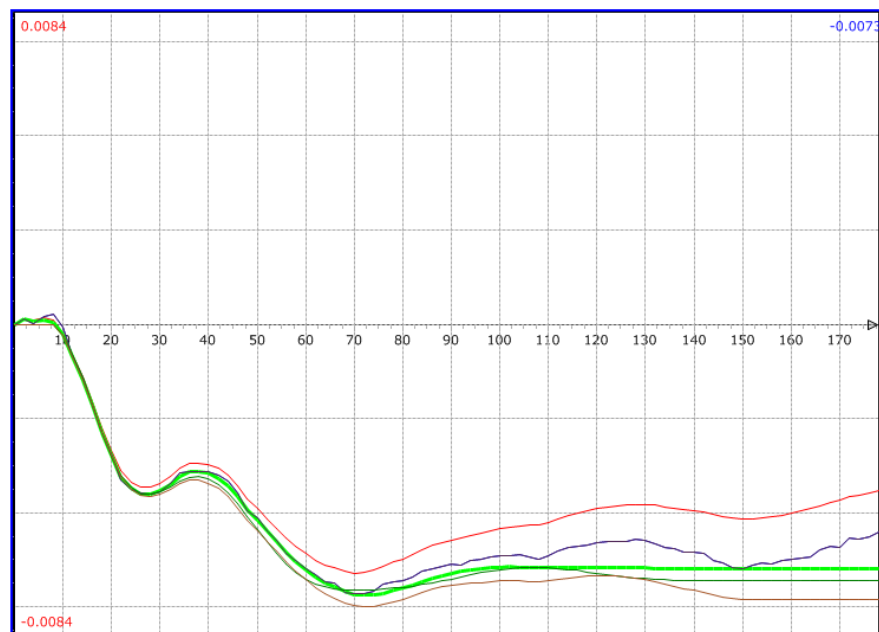
- **Standard FIR Algorithm:** Both smoothed and unsmoothed, with (or without) gain, dead and TTSS constraints, plus Null constraints (no model curves).
- **Transfer Function Method:** We included a new method that generates smooth transfer functions by converting the FIR models into low order transfer functions with pure dead time. This is an effective smoothing algorithm that imposes some degree of process realism on the model curves. It uses statistics to determine model order and tries to figure out from the step test data what degree of model complexity is warranted for the final model curve. If there is not enough evidence to support a complex high order model, it will give us a first or second order model. We can selectively add our own constraints, or override the auto-determined results. It means we can now properly handle long dead times in product quality models.
- **A New Quadratic Polynomial Based FIR Approach:** Essentially, we determine the model uncertainty "band" around the model curve, detect dead time and TTSS, and then determine how many straight lines it will take to approximate the Impulse Response without violating the uncertainty band. Once the impulse response is integrated into a step response, the straight lines in the Impulse Response become quadratic polynomials in the Finite Step Response Model. As a result, the models are very smooth. We do NOT simply do quadratic interpolation of the step response curve: We let the Least Squares problem figure out the best fit given the constraint that the impulse response is now made up of a series of straight lines. As a result, the parameter count is very low (at least one order of magnitude less parameters than a standard FIR), so the problem is better conditioned numerically and unrealistic squiggles are removed from the model curves if there is not enough evidence to support their presence (based on the model uncertainty estimate). In practice, the front part of the model curve has very closely spaced polynomials, while the backend may well be made up of just one quadratic polynomial that will reach a rate of change of zero at exactly TTSS. There is the option that Null models can be detected automatically and removed: If the relationship is so weak that the uncertainty calculation indicates there may be no model present, it will impose the Null constraint and remove the curve. We can selectively turn this feature on per model curve.



Note that all three methods are shown here. The model shown in brown is a smoothed FIR result, the light green model was computed using the Adaptive FIR method, while the dark green model is a Transfer Function.

Summary

INCA Discovery 1.0 has sophisticated new capabilities to deal with difficult and noisy step test data sets. The user can impose process knowledge in the form of dead time, TTSS and gain constraints. These constraints are imposed as equality constraints via a QP solver. Three model identification methods are provided and these algorithms can outperform the standard FIR approach. The unsmoothed standard FIR method is provided to act as a reference case. In this case, the user can set the smoothing factor, and turn on or turn off the direct transmission term (the initial kick in the model curve). The Adaptive FIR method reduces the parameter count significantly and approximates the step response with a series of quadratic polynomials that will have a zero rate of change exactly at the TTSS value. An example is shown below in bright green:



Note that the Adaptive FIR method produced a very smooth model without introducing any offset or bias problems. The last polynomial starts at 75 minutes and reaches a zero rate of change at 180 minutes where it gets completely steady (flat). The Transfer Function Method often provides superior results when compared to the FIR approach, and models display a higher degree of process realism. The user has control of the model order search range, but the algorithm is fully automated, and will automatically select the most appropriate model based on the available test data using the Bayesian Information Criterion.

INCA Discovery represents a significant advance in the quest for improved model identification technology.

Appendix:

