

PSIG 0406

Automatic Tuning of Pipeline Models

J. P. Modisette, Atmos International

Copyright 2003, Pipeline Simulation Interest Group

This paper was prepared for presentation at the PSIG Annual Meeting held in Palm Springs, California, 20 October – 22 October 2004.

This paper was selected for presentation by the PSIG Board of Directors following review of information contained in an abstract submitted by the author(s). The material, as presented, does not necessarily reflect any position of the Pipeline Simulation Interest Group, its officers, or members. Papers presented at PSIG meetings are subject to publication review by Editorial Committees of the Pipeline Simulation Interest Group. Electronic reproduction, distribution, or storage of any part of this paper for commercial purposes without the written consent of PSIG is prohibited. Permission to reproduce in print is restricted to an abstract of not more than 300 words; illustrations may not be copied. The abstract must contain conspicuous acknowledgment of where and by whom the paper was presented. Write Librarian, Pipeline Simulation Interest Group, P.O. Box 22625, Houston, TX 77227, U.S.A., fax 01-713-586-5955.

ABSTRACT

There are always initially unknown or wrongly known physical parameters in any pipeline simulation. These will cause erroneous model results unless the model is calibrated to compensate for them. This is true for both online and offline systems. Common sources of difference between a real pipeline system and that system's description on paper include buildup of deposits (or liquids in a gas line), pipe corrosion, changing ground thermal properties, and the effects of small impurities on fluid properties. In addition, SCADA systems may give incorrect readings due to sensor drift, and there's also opportunity for human error in recording the physical characteristics of the system. This article examines the effects of different sources of error on the results of both steady-state models (which are the basis of much offline analysis) and transient models.

Pipeline models have used many different techniques to compensate for these errors. These techniques are all based on matching the model's results to historical SCADA data. In offline models, this is typically done using a manual trial-and-error approach, while online systems usually include some sort of automatic tuning for this purpose, either based on feedback or state estimation. A new tuning algorithm is presented as an alternative to the manual trial-and-error approach for tuning an offline model or performing initial tuning of an online system, and results of tests of this new approach are given.

INTRODUCTION

If an online pipeline model doesn't have a working automatic tuning system, even if the model was initially tuned to provide good agreement with SCADA, it will very likely become useless in a few weeks after the initial calibration. While a vendor can usually manually tune a model so that it matches the instantaneous performance of the pipeline at the moment of the tuning, the unmodeled physical parameters of most pipelines vary significantly and rapidly enough that the state of the model will quickly deteriorate if nothing is done to track this variation.

Because model-based leak detection requires a very accurate model, it is the application where poor calibration has the most obvious effect. However, even in offline steady-state analysis some tuning is required. The user depends on an offline model to produce physically accurate results in a wide variety of situations, some of which may never have been attempted historically. In an effort to make the physical parameters of that model as close to reality as possible, it's calibrated to match any available historical situations that have recorded SCADA measurements. This is usually done via a laborious, iterative process of running the model for a historical situation, adjusting tuned parameters until the model agrees with SCADA for that particular scenario, and then moving on to the next scenario in an attempt to get agreement in all recorded situations. (Of course, if there were historical data available for every situation of interest, then there would be no need of a model – the user could just look up what would happen in the list.) Since the modeler has limited time to devote to tuning the model, this manual process may be terminated before the agreement of the tuned parameters with the available historical data is as good as it could be.

It is very difficult to make a pipeline model that perfectly simulates all relevant physical processes. Some parameters of the system, such as the exact effects of DRA (drag-reducing additive), are poorly understood. Unknown environmental changes can have drastic effects on fluid properties: a few rainy days can quadruple soil thermal conductivity, which causes large variations in the temperature profile, and that in turn produces significant errors in the viscosity of crudes or the density of gases. Wax or condensate buildup and

subsequent pigging also have dramatic effects. In liquids, small impurities can drastically alter viscosity, and properties like bulk modulus are often only approximately known to begin with.

In online simulators with automatic tuning, this family of problems is typically handled by relatively rapid adjustment of two or three tuning parameters. The exact choice of which parameters to adjust and how to adjust them is faced each time a new simulator is created. The simplest algorithm is to tune these parameters after each model step with any available extra SCADA measurements beyond what is needed for model boundary conditions (e.g. downstream temperature, or flow measurements on a piece of pipe which also has upstream and downstream pressure measurements).

This article describes some common sources of error, as well as which of them can or cannot be compensated with automatic tuning. It then describes common approaches to hydraulic and thermal tuning in more detail, and presents a new approach which makes a more correct use of all available data. The various approaches are then compared for a variety of off-line tests using simulated sources of error.

SOURCES OF ERROR

In general, pipeline models are driven away from reality by three classes of effects: sensor problems, unknown physical properties, and operator mistakes. All of these problems can be compensated to varying degrees by automatic tuning.

Sensor Errors

Pressure, flow, and other sensors often drift over time. However, except in rare pathological cases this is a small effect compared with other sources of error because the drift is comparatively slow (a timescale of at least months). Almost any automatic tuning scheme can compensate for normal instrument drift. If rapid drift is seen in, for example, a flow meter, it should be examined for periodic daily variations - this may indicate that there is an unexpected thermal effect (e.g. the the measured stream or the meter itself is for some reason at the ambient temperature).

Offline models are a different matter. It is difficult to distinguish between a pressure sensor that has drifted significantly and a bad roughness value for the adjacent pipe, as will be seen below. For this reason, the best option for tuning away sensor errors in offline models is just to let that effect be included in any other tuning, so that other parameters like the roughness are adjusted to compensate for the sensor drift.

Bad Physical Properties

The wall friction of a pipe can change over time due to the buildup of wax and scale in liquid pipelines, or due to the

accumulation of condensate in some gas pipelines, which can also reduce the pipe's effective internal diameter. Condensate buildup can significantly alter the operation of some pipelines over a day or less. Corrosion is another factor that can also change the wall friction, but like sensor drift it generally produces a very slow change.

Drag-reducing additive (DRA) is increasingly used in liquid pipelines to reduce energy costs. The DRA manufacturer usually provides some very rough estimate of the additive's effect as a function of concentration, but these estimates are never very accurate: they are usually correlations computed in laboratory facilities which don't exactly duplicate the pipeline environment. In addition, DRA tests over short distances won't account for gradual breakdown of the DRA in the line. For this reason, many pipeline companies are now computing their own DRA coefficients not just for each DRA type and crude, but separately for each section of their line. As far as an online model is concerned, uncertainties in DRA properties look just like other sources of pressure drop error. They can be characterized by appearing and disappearing rapidly (when the DRA is turned on and off) in a batched line, and also as pumps are turned on and off downstream from a DRA injection point (as the operating pump destroys most or all of the DRA going through it).

So far the properties have all affected frictional head losses directly. In heavy crude lines and gas lines, thermal effects can also be very important. The largest unknown thermal quantity in a pipeline is usually what is going on in the surrounding soil. The ground thermal conductivity is a strong function of the soil moisture content, which depends on recent weather as well as recent pipeline operations (e.g., whether the line has been running hotter or colder than usual in the last few days). The thermal conductivity just outside the line can strongly impact the temperature profile in the line. If the pipeline is pumping gas, or especially if it's pumping a viscous liquid such as a heavy crude, these temperature uncertainties translate into large hydraulic effects.

Operator Mistakes

Many pipelines do not have complete automation of all data used to drive the model. The two main examples of this that the author has seen are a number of systems where fluid properties (viscosity and sometimes density) are manually input, and one older gas pipeline where there was no automatic valve state information available in SCADA for certain valves. Since the model needs to know this information, in both cases it had to be manually input, and in both cases it was frequently manually input wrongly.

Manually input viscosities on a batched liquid system is an example in which automatic tuning CAN compensate for operator error. It is a common sight on liquid pipeline leak detection systems to see the hydraulic tuning struggling to compensate as an incorrectly identified batch fills the line, and then struggling to get back to the correct values as the batch exits. This distinctive signature says "something is wrong with this batch," and that can be the basis for automated

compensation of the error.

Incorrect valve states are difficult to compensate for automatically. In the case mentioned above, the on-line model was made to try, one at a time, all possible openings and closings of the manually-input valves whenever the model exhibited sudden and strong disagreement with SCADA flow measurements near a valve. (The uninstrumented valves were all in compressor stations in that case). Pipeline models sometimes need custom developments like this - in practice, no automatic tuning system is going to catch everything.

Model Deficiencies

In theory, automatic tuning can be used to compensate model deficiencies as well. For example, lack of a ground thermal model will result in what the model perceives as a rapidly changing ground temperature; so this could be handled instead by automatic ground temperature tuning. Or, if the model neglects pipe expansion in a liquid line, this is going to appear to the model as if the bulk modulus of the fluid is wrong.

The downside of using automatic tuning to correct for these model problems is that it does not distinguish between changes caused by the neglected physical effect and changes caused by a leak. Therefore this will necessarily degrade the sensitivity of a leak detection system. Similarly, it usually won't produce the correct behavior in circumstances very different from those where the data used to drive the tuning was taken, which makes it dangerous for offline models as well.

These problems can be remedied to some extent by allowing the automatic tuning to vary model parameters only gradually. Since the effects of a leak appear over a short time, the slow tuning then won't tune the leak out; once the model has detected the leak, the automatic tuning should then immediately be disabled to prevent it from compensating for the leak. Most of the types of input error mentioned above can be dealt with adequately with slow tuning. The main example that cannot be handled by gradual tuning is an incorrect bulk modulus (or, equivalently, failing to account for pipe expansion). The effects of bulk modulus errors are visible when there are large transients in the line, and they occur over the same timescale as these transients.

EFFECTS OF INPUT ERRORS

Some sources of error are more important than others; the important effects vary from pipeline to pipeline. The results of tests on a simulated pipeline to determine the magnitude of the effect of errors in various different physical parameters are given below. All tests were performed on a 30-mile pipe with an 18" internal diameter, and with upstream and downstream pressures of 800 psia and 300 psia respectively. The inlet temperature is 80° F and the ground temperature is 65° F. The mass, momentum, and energy equations for the fluid are solved self-consistently. The model uses a constant heat loss coefficient to ground and a fixed ground temperature; the heat

loss coefficient is computed for dry soil.

The tests compare the flow rate that would be computed with an accurate model to that which would be computed with a model with various errors in physical parameters. Tests were run for gasoline, for a crude oil with a viscosity of 300 cSt at 60° F, and for methane gas. For the methane cases, the error in linepack that accompanies the physical parameter errors is also shown; there was no meaningful linepack error in any of the liquid cases. The magnitudes of the physical errors were chosen to be consistent with those that the author has seen in the field for each property.

An offline model used to compute steady-state power consumption or steady-state capacity can expect errors comparable in magnitude to the flow errors listed in Table 1. For small values of the input error, the output error magnitudes can be expected to scale linearly with input errors: so, for example, a 2% inner diameter error in the methane case (presumably due to condensate) would produce a 5% flow rate error and a 4% linepack error. The viscosity input errors used above are so large because small concentrations of certain impurities, like detergents, can have an overwhelming effect on viscosity. The crude oil flows right at the edge of the laminar-turbulent transition, which is why the roughness change has so little effect on its flow rate, and why the viscosity has such a strong effect.

Steady-state errors can be quite different from the transient errors from the same sources. A relatively small error in modeled packing rate (the flow into the line minus the flow out) can have a strong effect on leak detection. If an online model generates packing rate errors at the level of 1% of mainline flow for 5 minutes whenever a pump starts, then it's not going to be able to detect a 1% leak in less than several multiples of 5 minutes. Packing rate errors caused by various input parameter errors in different circumstances are listed in Table 2.

The two methane cases with ground heat flow errors exhibit significant corresponding packing rate errors. The solid line in Figure 1, below, illustrates the packing rate error over time for the case with an incorrect ground temperature. The discrepancy in packing rate is mostly due to the incorrect fluid temperature and resultant incorrect density caused by the error in the treatment of heat loss to ground. Because of this, strictly hydraulic tuning cannot do much to compensate this error. This is also illustrated in Figure 1: the other three lines show three different attempts to correct the packing rate error by tuning model parameters.

The three different tuned parameters were: an offset in the downstream pressure measurement; a multiplier on the pressure gradient in the momentum equation; and a multiplier on the flow terms in the momentum equation. In each case the parameter was tuned to produce the correct steady-state flow given the initial upstream and downstream pressures. The pressure-gradient and flow corrections did largely compensate the packing rate error during the initial 3-4 minutes after

compressor start while the pressure surge was moving through the line. However, during the next 30 minutes as the slug of gas heated by volumetric compression during the compressor startup moved through the line, this type of correction didn't help. The correct temperature profile ("Dry soil") is compared with the uncorrected temperature profile with erroneous soil thermal conductivity ("Wet soil") in a snapshot taken 10 minutes after compressor start in Figure 2. The magnitude of the temperature surge has clearly been reduced in the case with greater ground thermal conductivity, as would be expected. This surge is not much affected by any of the tuning methods that solely adjust the momentum equation: it is necessary to tune thermal properties directly to capture this type of effect.

In addition to the two methane cases just described, Table 2 also shows a significant packing rate error caused by an erroneous viscosity in the case of the heavy crude oil. This occurs because this system is on the edge of the laminar-turbulent transition and decreasing the velocity brought the model firmly into the turbulent regime. The trend of packing rate over time is shown in Figure 3 for several different viscosities (the error reported in Table 2 is the difference between the 300 cSt and 200 cSt results). The qualitative differences between the 300 cSt and 200 cSt lines occur because in laminar flow (at 300 cSt) there is less dissipation of pressure waves, and so the shock of the pump start produces more and stronger oscillations as it travels up and down the line. Once the flow is firmly into the turbulent regime, a viscosity error produces little difference in the packing rate: see the 100 cSt and 50 cSt plots in Figure 3, which have a peak packing rate error of only 0.1% of main-line flow.

The exact behavior of a pipeline in the laminar-turbulent transition region is difficult to model exactly. In this model, a smooth interpolation using a cubic function was applied between Reynolds numbers of 2500 and 5000, with the Colebrook-White friction factor above that.

TUNING TECHNIQUES

Principles of Tuning

Tuning must always be driven by extra physical data beyond that needed to run the model. At its core a pipeline simulation is solving a set of differential equations, and so needs boundary conditions to complete that solution. These boundary conditions are pressure, flow, and/or temperature measurements. An actual pipeline SCADA system usually has far more sensors that are needed to drive the model; that additional instrumentation can be used to drive tuning. If a straight pipe has an upstream and downstream pressure measurement, that's sufficient for the model (with a temperature measurement somewhere also required if a thermal model is included), so anything beyond that (such as flow measurements) can be used for tuning.

Some set of parameters of the model must then be tuned. These are usually parameters that are considered to be constant during the solution of the model equations. Either physical quantities like the pipe roughness, soil thermal conductivity, or pipe internal diameter can be tuned, or "fudge factors" such as multipliers on pressure gradient or flow can be tuned. There is no generally accepted set of parameters to tune.

A typical tuning scenario for a straight pipe with pressure, flow, and temperature sensors at the upstream and downstream ends would be to use the upstream and downstream pressure and upstream temperature measurements to drive the model, use the upstream and downstream flow measurements to tune a "fudge factor" multiplying the pressure gradient, and use the downstream temperature measurement to tune the heat transfer coefficient between the fluid and the ground.

Tuning One Step at a Time

The simplest way to use extra SCADA data to drive tuning would be to solve for the values of the tuning parameters that produce model agreement with that data. In practice this isn't a good technique because it doesn't account for the fact that most of the processes that require tuning in the first place occur over much longer times than a single model time step. This means that in reality, the tuning parameters on one step are very close to what they were on the previous step. The tuning system can use this assumption to keep sensor noise from causing spurious changes in the tuning parameters.

One approach is to only adjust the tuning parameters a small fraction of what would be needed to get perfect model agreement with SCADA on each step. This type of feedback system can in theory be complicated to analyze, but actually the updating is almost always done very slowly, such that the tuning factor is only adjusted from one part in hundred to as little as one part in a million on each model step. This slow updating means that the system will be stable and well-behaved. If the model time step is t_1 and the time over which the tuning parameter is expected to change significantly is t_2 , then an update fraction of t_1/t_2 will give the correct behavior. In a leak detection system with automatic tuning, if the tuning update adjustment is too large then the system will tune out leaks. If it's too small, it will fail to tune out natural processes and generate false alarms. The place where "too large" and "too small" meet gives the leak sensitivity of the system.

This algorithm, which I will call the "step-at-a-time" tuning system, has one major limitation: it cannot easily tune two parameters that have similar effects. Pipe roughness and liquid viscosity would be such a pair of parameters: on any given step, either the roughness or the viscosity could be adjusted to make pressures and flows agree with SCADA. Since the system can't distinguish between the two, it has to pick one and tune that.

The only exception to this is when the two parameters are known to vary on extremely different timescales. For example, both the ground temperature and soil thermal

conductivity might be tuned to the downstream temperature measurement by updating the ground temperature, which varies on a 24-hour cycle, relatively quickly, and varying the soil thermal conductivity more slowly. Assuming that the ground temperature really does change much faster than moisture content, both quantities can be reproduced correctly.

Tuning Multiple Steps at Once

The type of tuning described above looks at one model step at a time, and therefore cannot directly benefit from characteristic patterns in the calculated tuning parameters over time. In the example mentioned earlier of a batch with an incorrect manually-input viscosity in a liquid pipeline, this system would adjust the tuning parameter continuously as the batch filled the line, and then adjust it back as the batch left the line. However, a human operator can recognize this pattern and realize what the problem is. A tuning system that can do the same would be useful.

This suggests that rather than trying to solve for the tuning parameters based on the extra instrumentation from just one step, the automatic tuning system could try to come up with a single set of parameters that best explain the tuning results from a series of steps. This can potentially work as long as the tuning parameters change slowly enough in reality that the values of the parameters are approximately constant for the set of model steps used to tune them.

Ideally the full set of nonlinear partial differential equations of the transient pipe model should be used to find the best-fit tuning parameters. This is mathematically difficult because the equations are nonlinear and are solved in most pipeline models either by linearization around the solution at the previous step, or by other techniques like the method of characteristics that also require the steps to be solved sequentially. If the tuning parameters change, then the solutions for early steps change too, which means that the linearized equations themselves for later steps change. This means that standard fitting techniques can't be used for the tuning parameters in conjunction with the full set of transient model equations, because the linearized equations for multiple time steps cannot be written down all at once.

In many pipelines, especially liquid systems, the line is often in hydraulic steady state. This is the key to the method proposed here for solving multiple states at once: only simultaneously solve for those steps where the line is steady. This unfortunately limits the applicability of this method in gas pipelines.

Instead of tuning to match the extra instrumentation from a single step, a system using this technique tries to match the extra data for many steps at once. To some extent this eliminates the limitation of step-at-a-time systems that they cannot simultaneously tune multiple parameters which produce similar effects. For example, such a system could simultaneously tune the viscosity of each batch and the pipe roughness in a batched line. The weakness of this system is that, even using multiple steps in this manner, it might be

difficult to distinguish different types of problem. This is investigated below.

This method can also be used to tune offline models. Historical SCADA snapshots for various different operating circumstances can be used to come up with a set of pipe roughness values and/or other tuned parameters that are consistent with what has been seen before and hopefully therefore are physically correct. Note that there is no guarantee that the one implies the other.

MULTI-STEP TUNING RESULTS

It was seen in Figure 2 that it is necessary to tune a thermal parameter independently of the hydraulic parameters, at least in gas pipelines. The author has found that for most online systems tuning one thermal parameter, such as the ground thermal conductivity, and one hydraulic parameter, such as a multiplicative factor on the pressure gradient in the momentum equation, works as well as any other set of parameters. If the model tuning occurs every time step using just the extra SCADA data from that step, there usually isn't enough additional data to tune more than one thermal and one hydraulic parameter. For instance, consider a straight unbroken pipe with pressure, temperature, and flow meters at the upstream and downstream ends: the two pressure meters and the upstream temperature meter are necessary to provide model boundary conditions. That leaves the downstream temperature meter to drive thermal tuning (the tuning will adjust the thermal parameter so that the downstream modeled temperature matches the measured temperature) and the flow meters to drive hydraulic tuning.

At each step the tuning algorithm updates the tuned parameters a tiny amount in the direction that will move the modeled upstream and downstream flow and downstream temperature values closer to the current measured values. This "tiny amount" is typically configured by the vendor when the online system is installed.

This method of tuning doesn't take advantage of all available information because the tuning algorithm looks at each model step as an independent occurrence from every other. If one of the compared quantities such as the modeled flow rate starts drifting away from the measured value, this approach doesn't take advantage of the context in which it happened: for instance, perhaps a new batch just started entering the line when the problem occurred. That suggests to a human user that the problem is with the fluid properties of the batch, rather than the friction factor.

Below, a new approach is examined that tunes to the extra data from multiple time steps at once. The algorithm, which is conceptually straightforward, is to choose the values for a user-defined set of tuning parameters so as to minimize the sum of squares of differences between the modeled values and the "extra" values for a provided set of scenarios. In an online model, this set of scenarios would be chosen from historical SCADA snapshots when the model was in different situations.

Here, the scenarios used to drive the tuning are generated with a separate offline model using the “correct” values of the tuned parameters. These scenarios will be referred to below as “test cases.”

The effectiveness of this algorithm can be judged by its ability to produce good estimates of those “correct” values. Of course, if the search space of tuned parameters contains a set of parameters that produces an exact match for all of the data in the test cases, then the tuning algorithm will easily pick that set of parameters. The test of the usefulness of this algorithm in real-world situations is whether it can pick a good set of parameters when there are unaccounted-for factors messing up the results. In reality, there are various sources of noise and physical effects not covered by the model even in the best circumstances. A tuning algorithm would ideally choose tuned parameter values that match physical reality closely in spite of these effects.

In this case, since these were offline tests on artificial pipelines (and even if they were online tests, it would have been prohibitively expensive to ascertain the “correct” values of the tuned parameters) the “noise” was created artificially by choosing a search space of tuning parameters that did not include the exact solution. For instance, if the exact pipe wall roughness was 0.0015", then the search space might include 0", 0.001", 0.002", and 0.003" for possible roughness values. If the tuning system cannot identify that the 0.001" and 0.002" are the best values for the roughness, that suggests that it wouldn't be able to pick the correct roughness values in the presence of other perturbations such as might be expected on a real pipeline. This is a preliminary test of this method; to test it thoroughly, it would need to be used on a real system and the results would need to be tracked over a long period and over different operating circumstances to verify that they produce a good fit to SCADA in all situations.

In most of the tests below, this algorithm found the best-fit values of tuning parameters by exhaustive search. In some cases, the search space of possible parameter values is too large for that; in those instances, the algorithm used simulated annealing to find the optimal set of tuned values. The simulated annealing produced exactly the same answers as the exhaustive search in all of the cases where exhaustive search was used. Only the final case used simulated annealing exclusively.

Figure 4 illustrates a simple test case: a gasoline pipeline where the algorithm tuned the batch density and the pipe roughness simultaneously. This is a plot of the best solutions for the tuned parameters that were selected by the algorithm. The tuning is driven by steady-state scenarios run at five different flow rates varying over a factor of three, so the solutions are chosen to provide the best fit to these five scenarios.

For each of the ten best solutions that were chosen, there is a square drawn in the figure. The position of the square indicates what the solution was: the x coordinate is the percent

error in the chosen density compared to the true density, and the y coordinate is the percentage error in the pipe roughness as compared to the true value. The area of each square is a measure of how good a fit that solution is: the area is proportional to the inverse of the rms error for all five test cases. Thus, a solution with a very large error will appear on the graph as a point, while a good solution will appear as a large square.

If the squares are about the same size visually then the algorithm doesn't have much preference between them. The origin of the graph is the “correct” solution. There is no square exactly at the origin because the mesh of points the algorithm searched for the correct solution deliberately did not include the origin for the reasons described above. The search space covers the entire displayed area of the plot.

In Figure 4, the chosen solutions are spread out over quite a range of roughness and density values. The tuning is clearly unable to distinguish between roughness error effects and density error effects: if it increases the density and simultaneously decreases the roughness, or decreases the density and increases the roughness, the results are about as good a fit.

In Figure 5, the analogous case for a methane pipeline is shown. Here the roughness and inside pipe diameter are varied, reflecting an unknown amount of condensate in the pipe. The roughness is varied over the entire y axis, while the diameter is varied from 95% to 105% of the correct value. All of the ten best answers chosen by the algorithm has the pipe diameter as close as possible to the correct value. However, there was still no strong preference for the correct roughness. Again, we see that to some extent the roughness error could compensate for the diameter error.

The original intent of this algorithm was that by using data from a wide variety of operating scenarios, it could zero in on the correct tuned parameter values. Evidently, just varying the flow rate isn't enough to distinguish exactly where the errors lie. In Figure 6, a liquid line was tuned with two batches (one gasoline and one diesel) flowing through two pipes with different diameters (16" and 18"). The density of each batch and the roughness in each pipe were tuned. There were twenty test cases used to drive this, representing each batch in each pipe at each of five different flow rates. With four tuned parameters it becomes a little difficult to visualize where the chosen solutions lie, so unlike in Figures 4 and 5, the sum of magnitudes of roughness errors over the two pipes and the sum of magnitudes of batch density errors are used as the axes (instead of the values of those errors). This means that there can be multiple distinct solutions that appear on the plot as if they were at the same point: for instance, a solution with +1% roughness and +1% density would appear the same as a solution with -1% roughness and +1% density, because the plot shows sums of squared errors. Again, the correct solution is at the origin (which is in the lower left now) and is not included in the search space. The results in Figure 6 are better than those from the previous cases, showing a definite

preference for the solutions as close as possible to the correct values, but still producing a number of solutions quite far from the correct value that have similar quality numbers.

Figure 7 shows the results of the same runs with inclined pipes. The first pipe has a 1% grade uphill and the second pipe a 1% grade downhill. This should help the algorithm differentiate between density errors and roughness errors, and indeed the results are a little better, with most of the weight in solutions as close as the search space allows to correct.

Finally, Figure 8 is a test with five different batches (two types of gasoline, two diesels, and one of water) running through five different pipes. This time only one flow rate is used instead of the five in the previous cases. Here we see that 9 of the top 10 solutions are as close as possible to the correct values. This suggests that with enough data, it's possible to distinguish between fluid property errors and pipe roughness errors.

CONCLUSIONS

It is important to tune to compensate for certain types of errors in input data. In gas pipelines it's vital that an accurate model have a good representation of the ground thermal properties. In liquid pipelines this is only true of crudes with viscosities that are strongly affected by temperature. These results hold for both steady-state tuning of offline models and also tuning of online models so that they produce correct packing rates for use in leak detection. Errors in other parameters can also produce significant errors in the hydraulic results.

It is hard to distinguish the hydraulic effects of an incorrect roughness value from errors in other hydraulic parameters like the fluid density or viscosity. Given this, it is probably not worthwhile to tune these parameters independently of each

other unless a large amount of data is available to drive the calibration. For step-at-a-time tuning of online systems, there is likely no advantage to tuning more than one hydraulic parameter for a given region between SCADA measurement points. This may also be true for thermal parameters, but that wasn't investigated in this article.

The automatic tuning algorithm presented above performs well for liquid systems only when fed data from many different scenarios. Specifically, it can distinguish errors in different physical parameters only when scenarios are available with different batches moving through different pieces of pipe. This data should be available for most batched liquid lines. This approach can provide significant labor savings and improved quality of tuned results for offline models or initial calibration of online models of such pipelines.

Since gas lines don't have batches, it is probably only applicable to gas lines when there aren't very many physical parameters in question. In that case, there's no reason it would perform better than the step-at-a-time method. It could however still be useful for tuning an offline model.

ACKNOWLEDGEMENTS

The author wishes to thank Atmos International for support of his attendance of the PSIG conference, and also Energy Solutions International for support of the early phase of this work. He is especially grateful for the help of Dr. Greg Morrow of the latter company regarding automatic tuning of online models. He also wishes to thank Dr. Jerry Modisette for his input on the article.

TABLES

Fluid	Erroneous Parameter	Error Magnitude	Flow Error	Linepack Error
Gasoline	Pipe roughness	0.0002"	1.7%	
Crude	Pipe roughness	0.0002"	0.04%	
Methane	Pipe roughness	0.0002"	0.6%	0.06%
Gasoline	Specific gravity	1%	0.5%	
Crude	Specific gravity	1%	0.5%	
Methane	Specific gravity	1%	0.4%	0.0%
Gasoline	Ground temperature	5° F	0.001%	
Crude	Ground temperature	5° F	0.08%	
Methane	Ground temperature	5° F	0.3%	0.5%
Gasoline	Ground conductivity	Wet vs. dry (300%)	0.03%	
Crude	Ground conductivity	Wet vs. dry (300%)	0.7%	
Methane	Ground conductivity	Wet vs. dry (300%)	0.6%	1.1%
Methane	Pipe inner diameter	1%	2.6%	2.0%
Gasoline	Viscosity	50%	3.2%	
Crude	viscosity	50%	13%	

Table 1 – Steady-State Flow and Linepack Errors

Fluid	Transient Scenario	Erroneous Parameter	Error Magnitude	Packing Rate Error
Crude	Soil change over 2 hrs	Ground conductivity	Wet vs. dry (300%)	< 0.001%
Methane	Soil change over 2 hrs	Ground conductivity	Wet vs. dry (300%)	0.03%
Methane	Compressor start ¹	Ground conductivity	Wet vs. dry (300%)	0.9%
Gasoline	Incoming batch ²	Specific gravity	1%	< 0.001%
Gasoline	Pump start	Specific gravity	1%	0.01%
Crude	Pump start	Viscosity	50%	5%
Gasoline	Pump start	Bulk modulus	5%	0.3%
Methane	Compressor start	Ground temperature	9° F	0.8%

Table 2 – Transient Packing Rate Errors

¹ Pump and compressor starts were simulated with a 145 psi pressure increase and accompanying 18° F temperature increase over 200 seconds.

² A batch started pumping at time zero with the erroneous fluid properties. Before that, the line was filled with a batch with correct fluid properties.

FIGURES

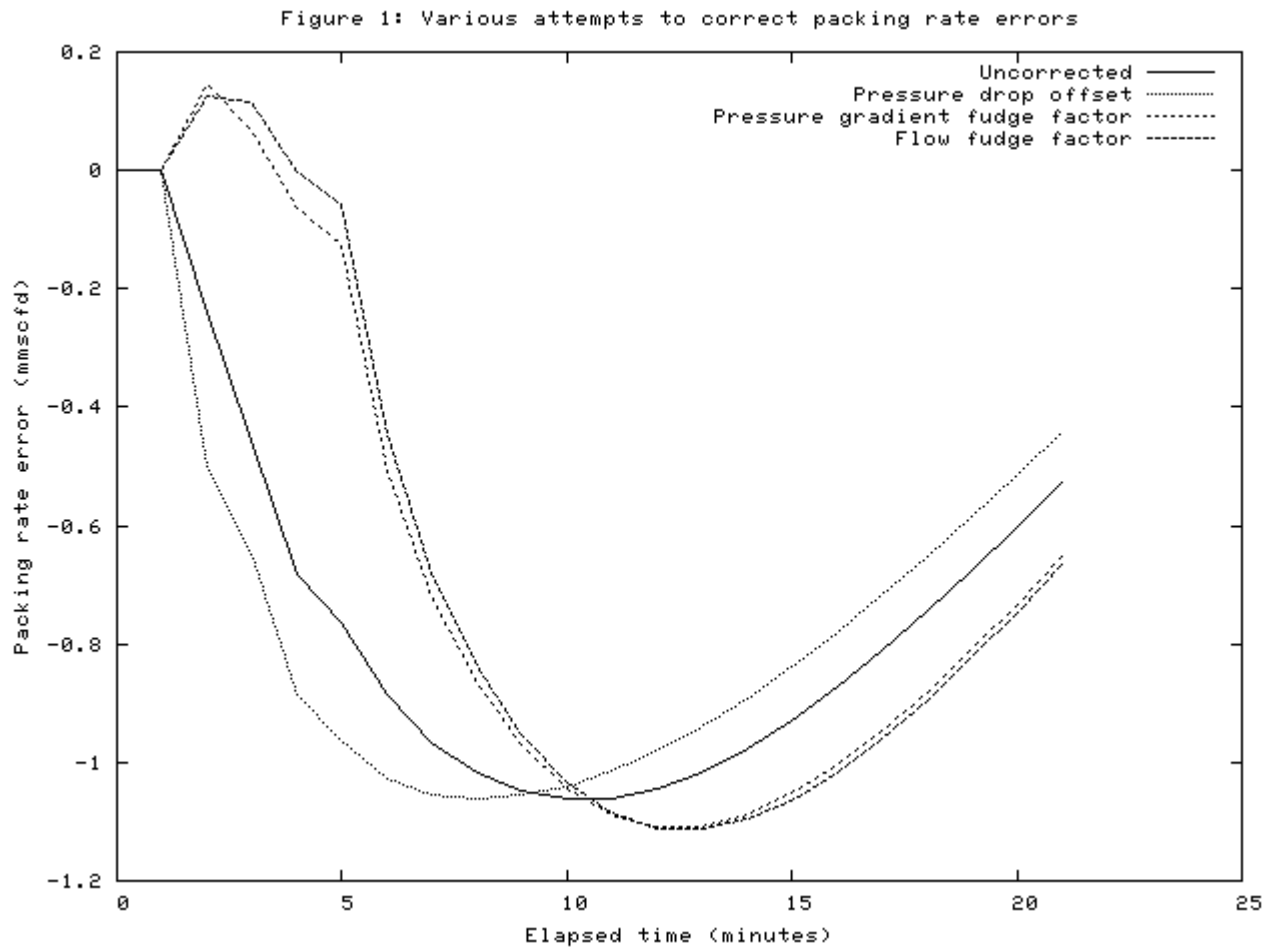


Figure 1 – Various attempts to correct packing rate errors

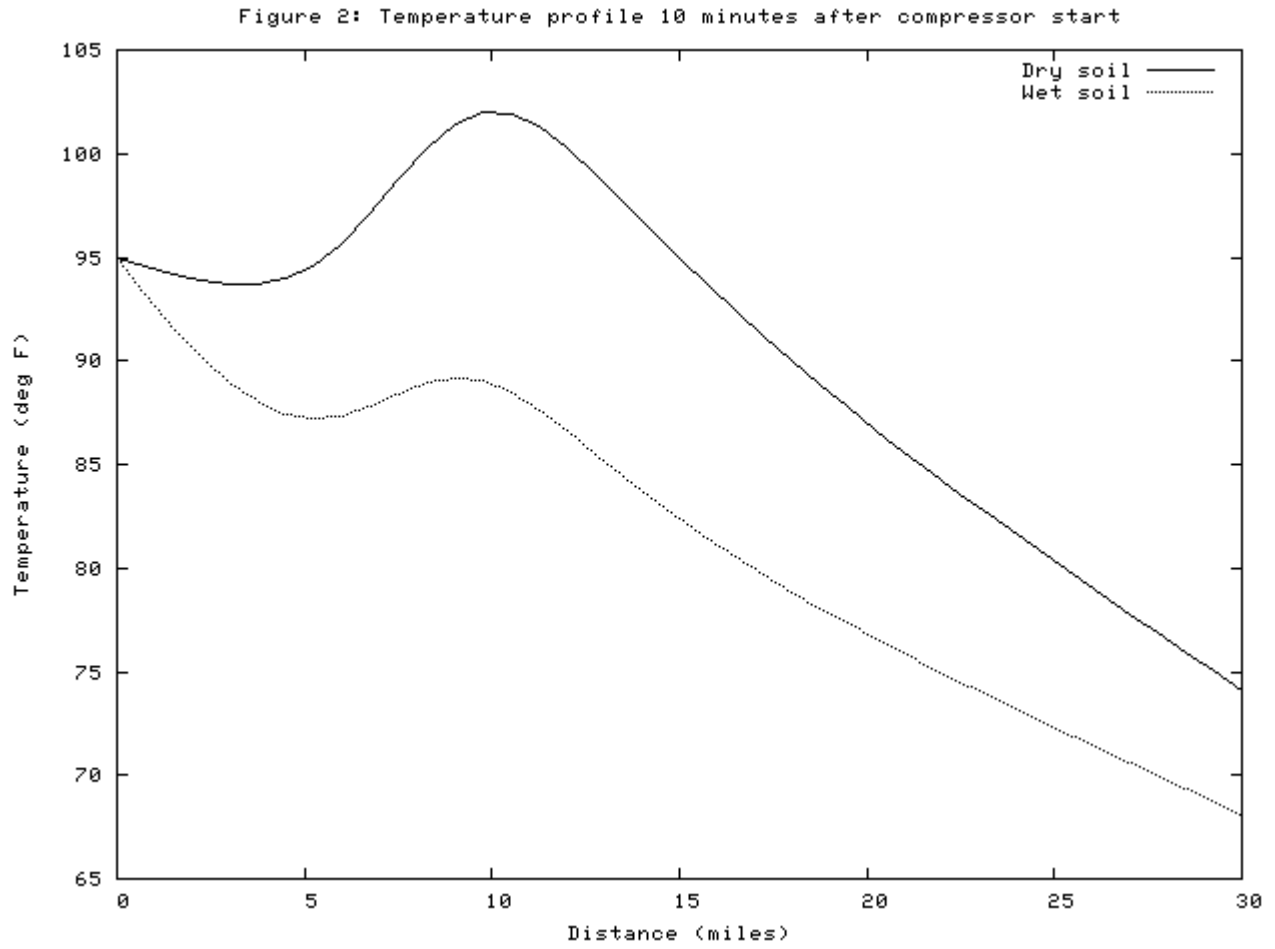


Figure 2 - Temperature profile 10 minutes after compressor start

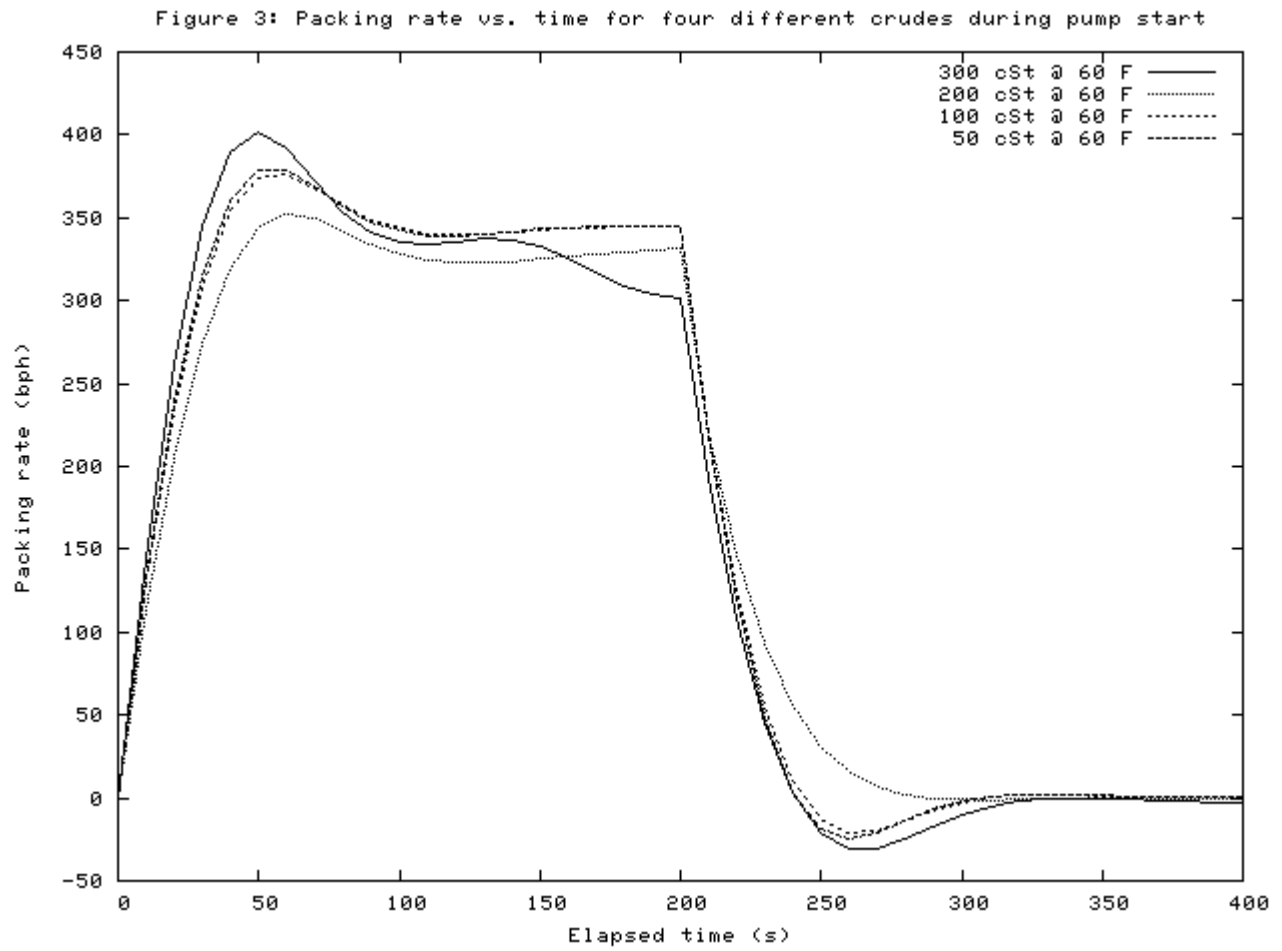


Figure 3 – Packing rate vs. time for four different crudes during pump start

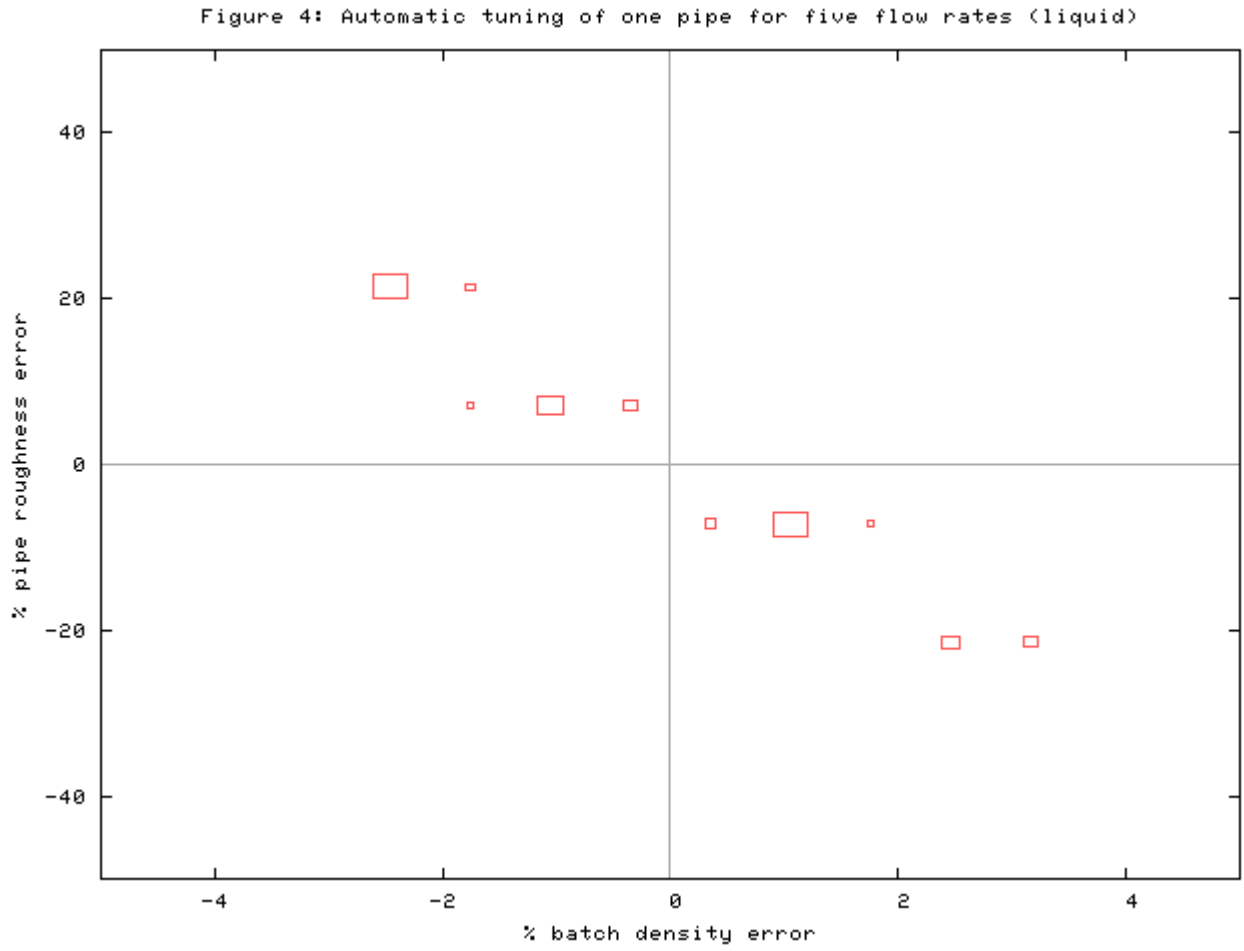


Figure 4 – Automatic tuning of one pipe for five flow rates (liquid)

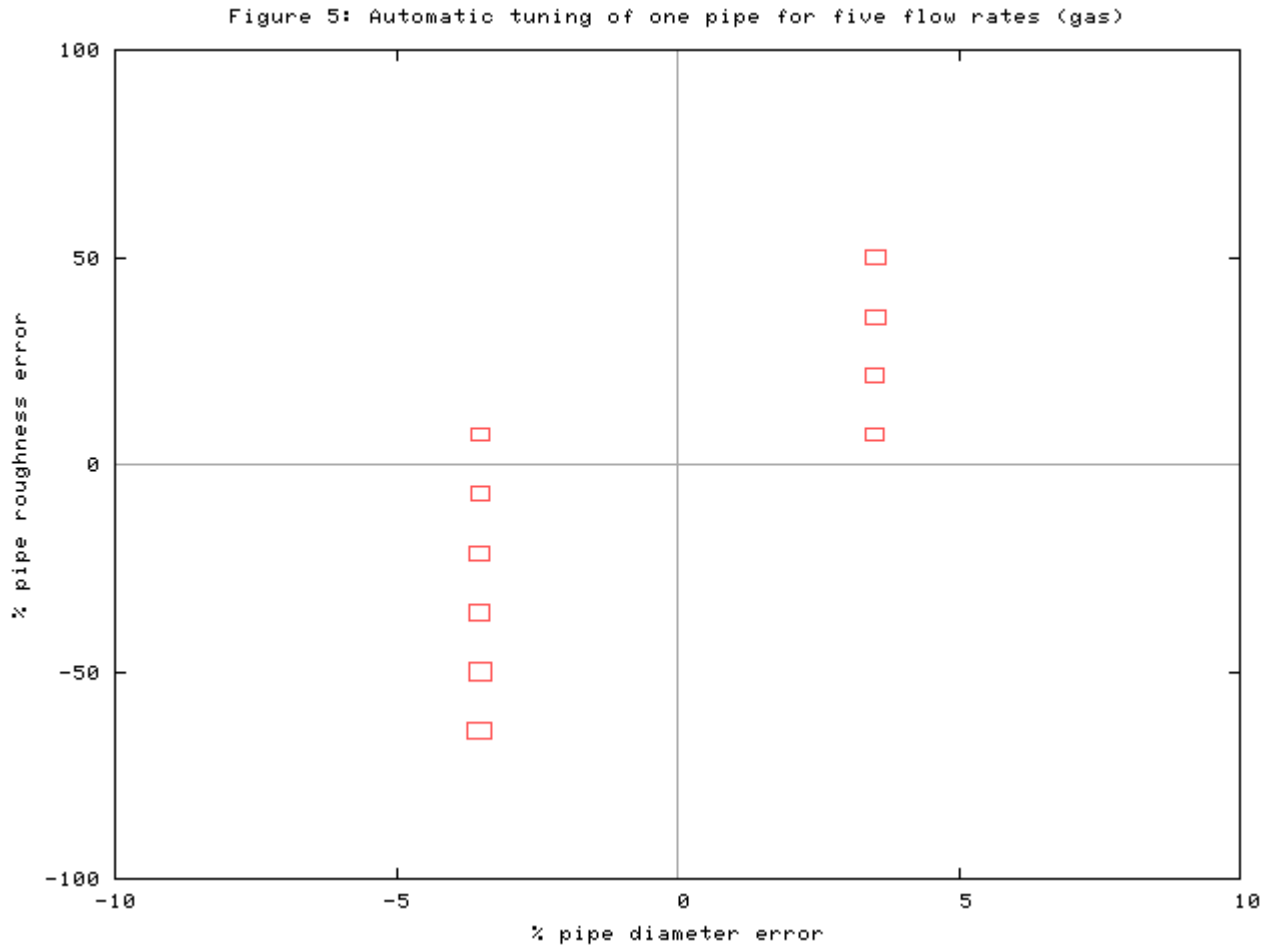


Figure 5 – Automatic tuning of one pipe for five flow rates (gas)

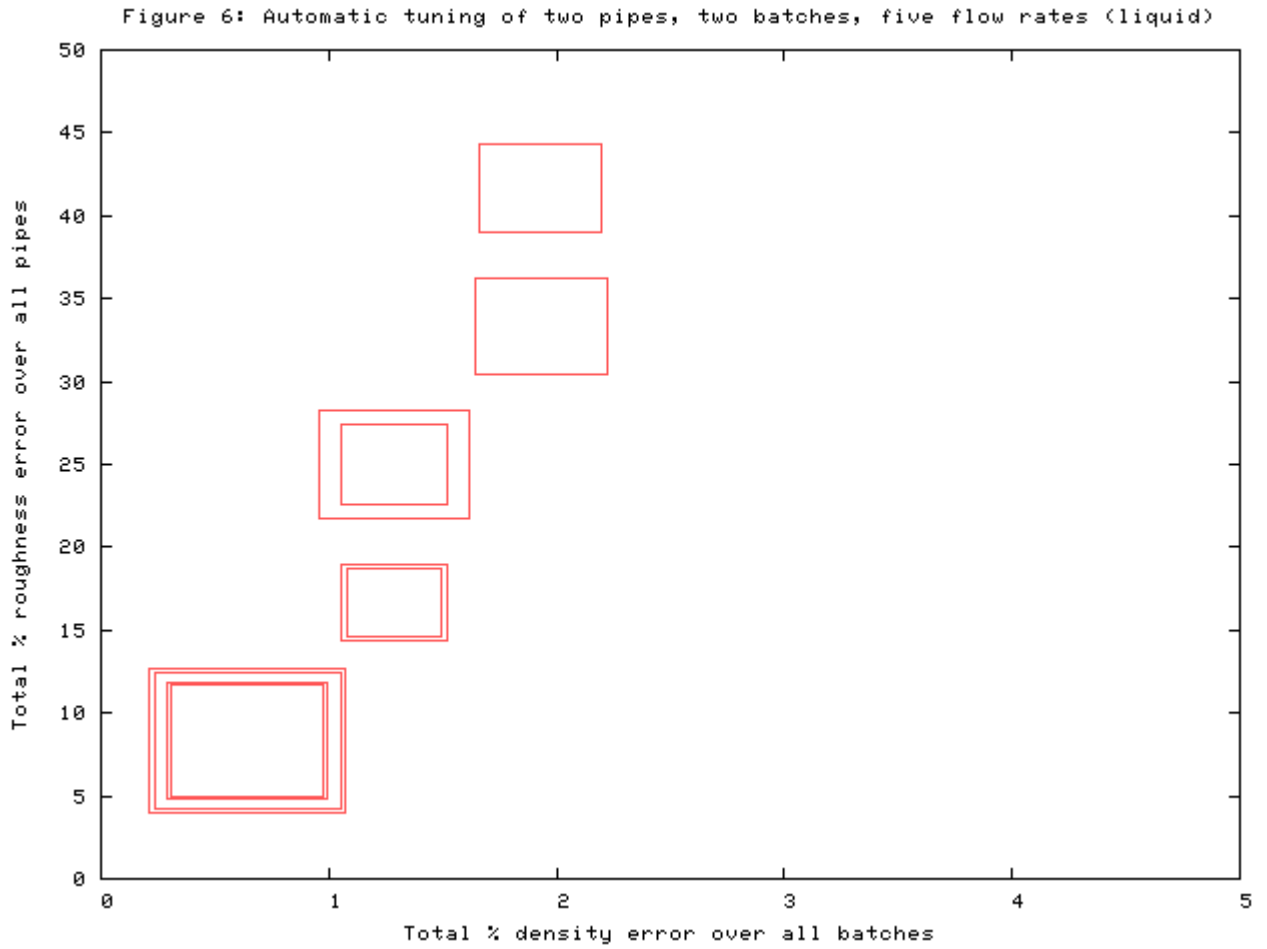


Figure 6 – Automatic tuning of two pipes and two batches for five flow rates (liquid)

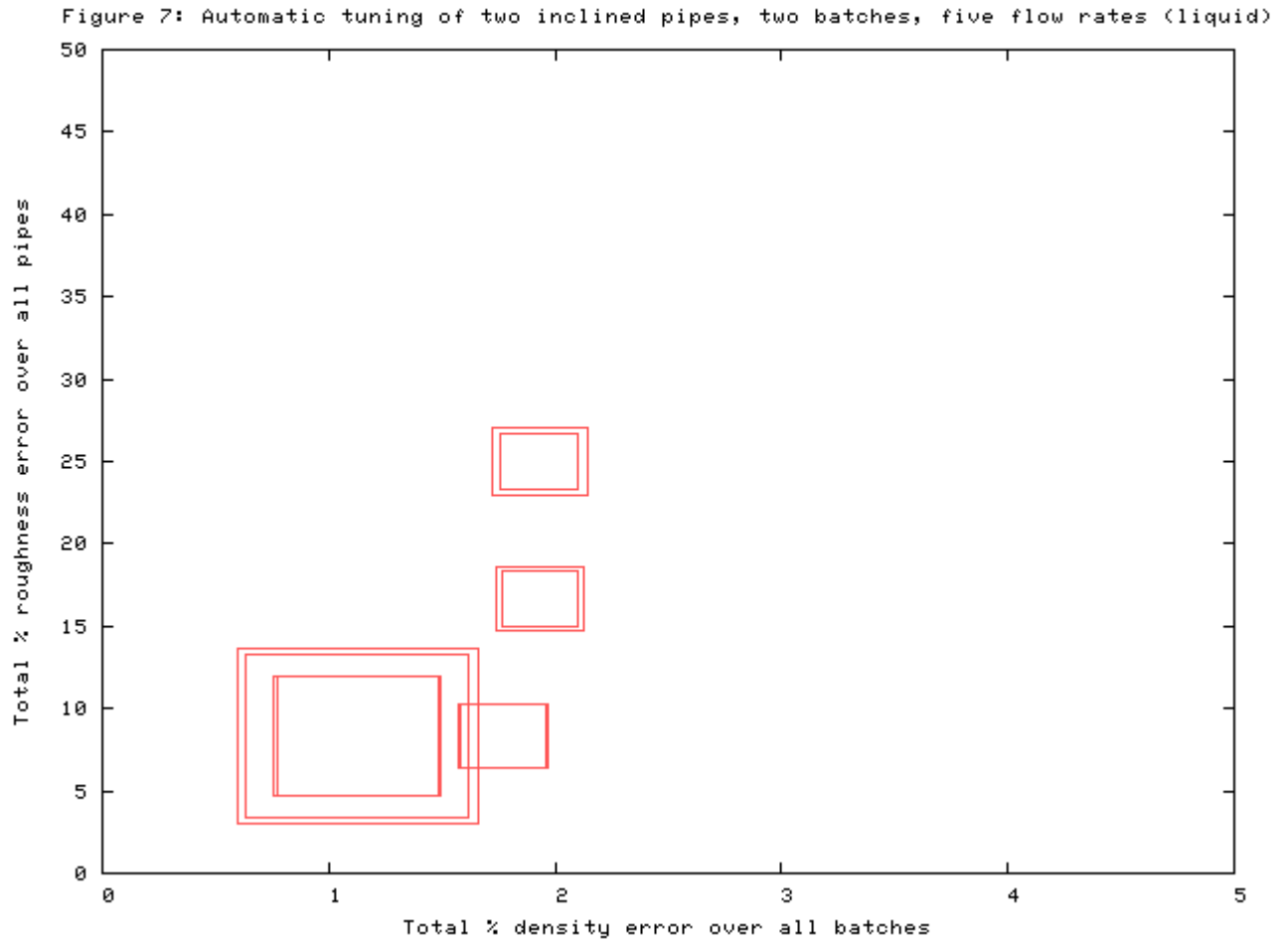


Figure 7 – Automatic tuning of two inclined pipes and two batches at five flow rates (liquid)

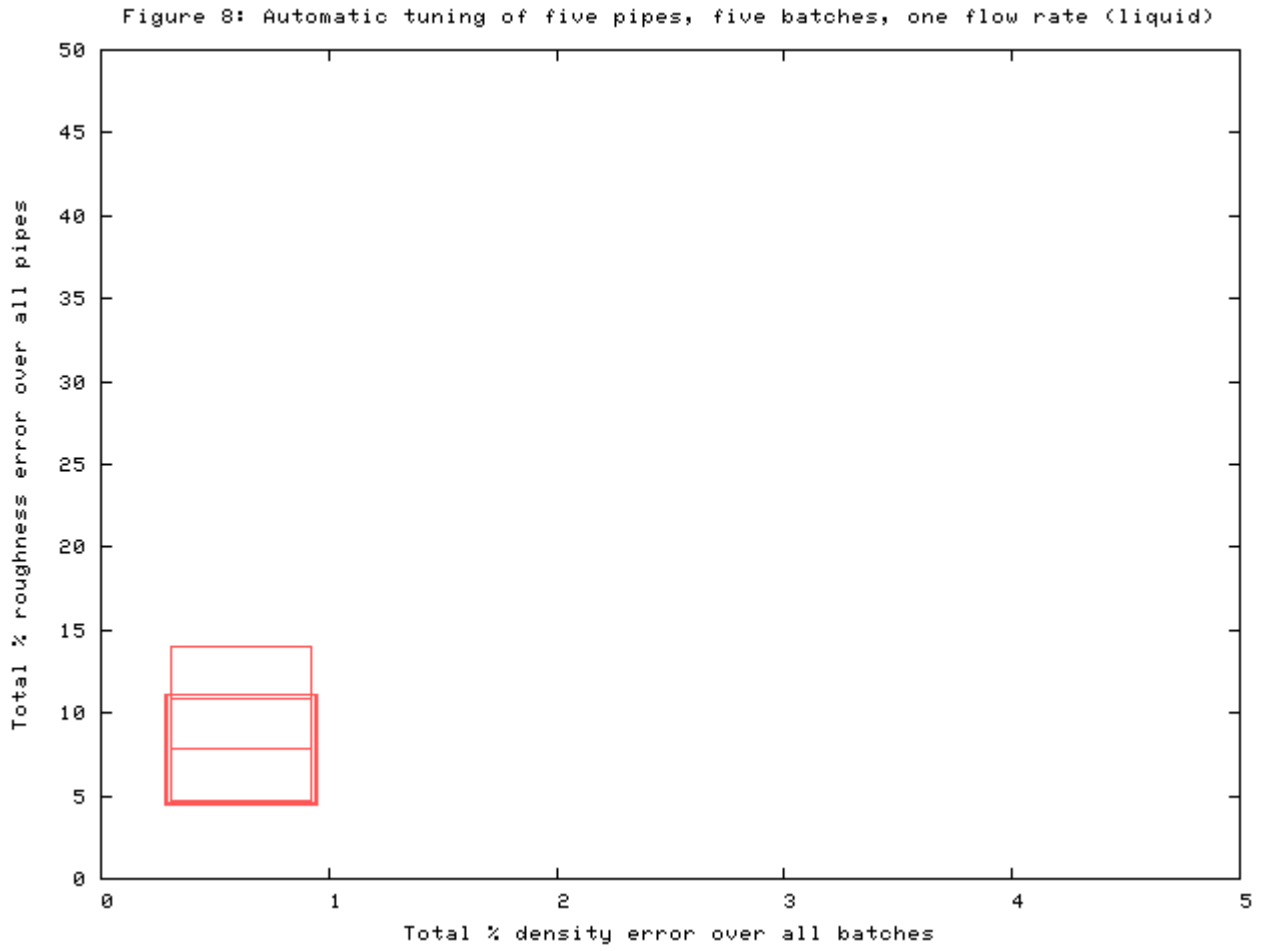


Figure 8 – Automatic tuning of five pipes and five batches for one flow rate (liquid)