Efficient Large-Scale Optimization Under Uncertainty with Multiple Proxy Models

N. H. Goodwin (Essence Analytics)

email [**info@essenceps.com**](mailto:info@essenceps.com), tel. +44 (0)7710 403 255

Keywords: proxy model, optimization, uncertainty, reservoir simulation

MSC: 90C06, 65K10

Acknowledgments. The author would like to thank Stone Ridge Technology for use of their Echelon reservoir simulator.

**Abstract**

The authors have undertaken over the last two decades many reservoir optimisation studies, including well locations, drilling schedules and flow rates. Details of proxy models for uncertainty quantification with history matching have been fully described [1], and their implementation resulted in the industry's first fully probabilistic forecasting tool. More recently, optimisation of intelligent well design under uncertainty has been illustrated with a case study [3]. The current work shows how multiple proxy models can be used to greatly accelerate large scale optimisation under uncertainty.

The parameters were controls for each well at each 3-month time step (total 1440 parameters). Separate studies were performed for rate and BHP controls. There was a total of 50 geological models representing uncertainty. All 50 models were used throughout the analysis.

The novel features include:

* partitioning of the objective into multiple proxy models
* encoding of categorical variables, particularly multiple static models, as an optimal mapping to integers
* Bayesian integration over uncertain variables

The proxy models are Gaussian process models with a linear regression component.

The overall process is two level. First it optimises the expected improvement using the proxy models, and then one or more simulation runs is performed at the chosen parameter values. Each optimisation function evaluation involves a summation, at the proxy level, over the static models.

The reservoir simulations are performed sequentially, one at a time, and there is no use of cloud or cluster computing. The simulator used is the GPU-based Echelon from Stone Ridge Technology.

The total number of simulation runs was under 3,000 (under 60 runs per geological model).

# Background

The authors have undertaken over the last two decades many reservoir optimisation studies, including well locations, drilling schedules and flow rates [4], in major projects in the North sea and Gulf of Mexico. Proxy models for history matching have been used extensively by many oil companies since 2001 in commercial software. Details of proxy models for uncertainty quantification with history matching have been fully described [1], and their implementation resulted in the industry's first fully probabilistic forecasting tool [2]. More recently, optimisation of intelligent well design under uncertainty has been illustrated with a case study [3]. The current work shows how multiple proxy models can be used to accelerate large scale optimisation under uncertainty.

The Olympus challenge is, indeed, challenging, and it was used to investigate an approach using sophisticated proxy models and with no limitations or approximations. The full set of modifications was used, and no dimension reduction was performed. Full proxy models were implemented using Gaussian processes together with a polynomial regression component. Single simulation runs were submitted, and a very modest hardware environment was used. Throughout the software package EssRisk was used, including a sophisticated results calculator, which was employed to perform all the npv calculations.

Some authors use a single proxy model to emulate the objective function. In contrast, we build a large number of proxy models to emulate the simulator response, and then build the objective function from these multiple proxy models. The difference in performance is significant, but it does require an efficient implementation of the proxy models. By constructing multiple proxy models, it is common to achieve a complex history match in under 150 reservoir simulation runs.

# Understanding the problem

This challenge has 18 wells and 80 time periods, so there are 1440 different control parameter values which must be optimized. There are 50 possible geological realizations.

We use oil production and water injection rate controls, and for comparison a separate study with BHP controls for producers and injectors.

The problem as posed, and which we attempted to solve, was to optimize the 1440 controls, averaged over all the geological models. The initial assumption is that a particular control value is used in all 50 geological models.

There are two major issues which need to be noted.

First, as more knowledge is discovered about the field, there will be a reduction in uncertainty and some geological models can be rejected as unlikely. We may be underestimating high side or low side potential, or worse underestimating, because the optimal values are based around a geological average. This may result in a decision not to develop an economic asset. In contrast, optimizing each geological model independently will give a better understanding of each model and the long-term economics. For example, an optimized low case may indicate an economically viable field. If the field is developed, and the low case turns out to be correct, then it would be operated using those optimized low case control values.

The second problem is the concept of active and reactive controls. The rate controls are initially active, but rate control will change to pressure controls (and vice versa) depending on the geological model. Hence, for a given set of control parameters in the optimization process, the actual controls will vary significantly across geological models. The controls have a significant reactive element. This introduces a high level of non-linearity and makes the optimization process much harder.

It is possible to consider other types of control which are explicitly, rather than implicitly, reactive. For example, we can optimize a policy “use maximum rates until water cut rises above 80%, then shut down for 30 days, then try to open again”. This is clearly reactive, and it could also be regarded as being more closely allied to the actual business problem. It is almost certainly an easier problem to optimise, as there are fewer optimisation variables.

In our work, we deal with active rate controls which are often in a reactive mode. We feel a completely reactive, or policy-based, optimization would be a very useful future exercise.

# Comparison with other approaches

There is unfortunately a myth that proxy models impose a severe computational burden. We hope that this study dispels that myth. We are using 80 different proxy models, and train them with over 2000 simulation runs. Each function evaluation during the proxy optimization process involved 80x50 = 4000 proxy model evaluations (80 proxy models times 50 geological models).

Early work on proxy models used data science languages such as R. We now implement proxy models directly in using multi-threading Java and have continuously tuned performance and memory over a period of ten years or more. The resulting improvement in performance over R is at least three orders of magnitude.

As has been described elsewhere [1], there are many varieties of proxy models, and it is important that the proxy model is interpolating. That is, it reproduces exactly results where a simulation has already been run. Pure regression models are not interpolating, and neural networks are neither performant nor interpolating.

Many optimization approaches in the literature have been based on genetic or evolutionary algorithms, including particle swarm and differential evolution. These have many attractions, as they are simple to code and can take advantage of parallel simulation resources.

However, these methods can easily become stuck in a local optimum, and reports of convergence should be taken with extreme caution. A genetic algorithm with a population size of 100 is not going to optimize a problem with 1440 variables in anything less than hundreds of thousands of simulation runs, and many more if there are multiple geological models. The behaviour of these algorithms tends to be that they find an improvement, but it can be many iterations later before further improvement is found, so a naïve practitioner may mistakenly believe that it has converged. It has not.

Adjoint methods have some attraction, as any information to help the optimization process should be grasped. However, as is well known in adjoint-based history matching, again there is the problem of getting stuck in a local optimum, so a stochastic element is necessary to avoid this. Of course, many simulators do not offer an adjoint solution, and even where they do, they are not able to calculate adjoints for all the variables of interest. In contrast, proxy models come with a derivative almost for free.

# Categorical variables and geological models

There are various possible approaches to handling discrete geological models with proxy models.

* Treat each geological model independently and have a set of proxy models for each geological model. The benefit is that the proxy model is more accurate for each geological model, but the total number of simulations increases significantly. For 50 geological models and n total simulation runs, each geological model is trained with only n/50 simulation runs.
* Use the geological model as a categorical variable and have a single set of proxy models across all geological models. We use modern data science techniques to handle categorical variables. A common approach is to introduce dummy variables and use what is called ‘one hot’ encoding. These are severely limited, particularly when the number of categories grows. A better approach is to transform the categorical variables into a mapping to integers.

The choice between these options depends on several factors, such as the computing environment, the number of available simulation licenses, and whether the geological models are similar enough that they can learn from each other.

The approach we take is to use the geological model as a categorical variable and have a set of proxy models common across geological models. We take this approach to minimize computing resources and take advantage of novel ideas on treating categorical variables. We automatically optimize the mapping to integers using a combination of different genetic algorithm operators, which include permutation and swap operations as well as ordered residual operations. This is performed during the subset selection procedure and has proven to be both very efficient.

# The overall approach

There are three phases to the approach. The first phase is a pure experimental design, using a Latin Hypercube design. We use 50 or 100 runs during this phase (one or two runs per geological model). The second phase is to submit a sequence of single simulation runs, with different geological models and control parameter values. The third phase is to submit batches of simulation runs corresponding to the 50 geological models, where each batch shares the same control parameter values.

The approach in the second phase is to use the proxy model to determine control parameter values, choose a geological model (using a round robin selection process), submit a single simulation run, and repeat.

* Update the proxy models
* Determine set of modifiers
* Choose geological model
* Submit simulation run
* Repeat

# Determine the set of modifiers

The control parameter values are determined based on the proxy model. After each simulation run, the process is:

* Update the proxy models
* Optimise the expected improvement

The proxy model is updated in two stages:

* Choose terms in the polynomial regression model and build a Gaussian process model
* Optimise the hyperparameters in the Gaussian process model

The polynomial regression model uses Bayesian averaging, and uses a modified version of AICc as a criterion for choosing terms [1].

In addition to selecting the terms in the model, we choose the mapping function for the categorical variables. The mapping is defined in terms of a permutation, and we introduce crossover and mutation operators for optimizing the permutation. The crossover operator is modelled on differential evolution concepts. In addition, we use a residuals operator, which calculates the best permutation based on the regression residuals.

The Gaussian process model uses a Matern covariance function, with different correlation lengths in different directions. These correlation lengths are optimized initially uses a genetic algorithm, followed by application of BOBYQA [6].

The expected improvement is a sum over all the geological models, and for each geological model there is a sum over all the proxy models.

If the current best value of the objective from simulation runs is and is the estimate from the proxy models then the estimated improvement is I =

The likelihood of achieving this improvement is given by the normal density function

(173)

The expected improvement is then

(174)

The expected improvement is optimized in two stages. First an evolutionary algorithm is used. Subsequently a limited memory quasi-Newton method is used,(LBFG-B [7]. The quasi-Newton method requires the derivatives of the objective function, which means that derivatives of the expected value and variance of the proxy models need to be calculated. This involves derivatives of the Matern functions.

For a given proxy model the estimate and variance are given by

# Overview of optimization methods used

There are a variety of optimization methods used throughout the approach. BOBYQA is a no derivative method and LBFGS-B is a bounded limited memory quasi-Newton method. For both hyperparameter tuning and optimizing the expected improvement, we start with a genetic algorithm, and then refine using local optimization methods. Hyperparameter tuning could, in principle, use derivatives of the restricted likelihood function, but this has not yet been implemented.

It is clear that LBFGS-B, where derivatives are available, is far superior to genetic algorithms. However we start with the genetic algorithm in order to increase exploration from reasonable starting points, under the assumption that the function being optimized has multiple local optima.

Table 1 use of optimization algorithms at different stages

|  |  |  |  |
| --- | --- | --- | --- |
|  | Subset selection | Hyperparameter tuning | Optimizing expected improvement |
| Genetic algorithm | Checkmark | Checkmark | Checkmark |
| BOBYQA |  | Checkmark |  |
| LBFGS-B |  |  | Checkmark |

In terms of convergence behaviour, we examine a variety of optimisation algorithms, as they all share underlying principles. We consider adjoint/quasi-Newton, ensemble methods, genetic algorithms (including evolutionary algorithms, particle swarm and differential evolution), and finally proxy-based methods.

In 3 dimensions, 3 data points defined a two-dimensional plane. Similarly, in hyperspace of 1440 dimensions, 1440 data points will define a hyperplane in 1439 dimensions, and smaller numbers of data points will define smaller dimensional hyperplanes. In directions orthogonal to that hyperplane, we have no information.

The danger in any optimisation algorithm is that we explore only on a small dimensional hyperplane, and never explore orthogonal to the hyperplane. By moving around the smaller dimensional hyperplane, we may thing we have converged, but, in reality, we have just explored a tiny fraction of the total space.

Worse, as the number of dimensions increases, all points tend to be equidistant from each other, and most of the space is near a boundary. These considerations put a major strain on all optimisation algorithms.

For the Olympus challenge, we have the choice to optimise directly on the simulator or to use a proxy model. Some optimisation algorithms build a local approximation as they proceed. In each case we are dealing with the same number of dimensions, and a similarly shaped objective function, depending on the quality of the proxy models.

For adjoint/quasi-Newton methods, we have the full 1440-dimensional derivative. In an unconstrained problem, with a fixed convex Hessian, quasi-Newton methods in theory take 1440 updates to generate the true Hessian, so 1440 function and adjoint evaluations would be required to converge.

In practice, these theoretical guidelines need to be considerably modified:

* the number of function evaluations goes up as the problem becomes more complex and where there are multiple global minima.
* in test problems, there is a lot of variation in the number of function evaluations divided by problem dimension, so guidelines are difficult to give a priori.
* the number of function evaluations goes down depending on the number of active constraints. It is more related to the number of free (inactive) variables.
* a limited memory version of quasi-Newton will increase the number of function evaluations

Experience has shown that on many optimisation problems, a reasonable convergence can be found in fewer than the theoretical maximum, but this is problem specific, and depends on a strong ranking of the importance of variables.

For ensemble methods, the population size of the ensemble should ideally be greater than 1440. Otherwise the danger is that we remain on the smaller dimensional hyperplane and converge to a sub optimum. The total number of function evaluations for convergence would be some multiple of 1440, depending on the number of iterations.

Powell's derivative-free algorithm BOBYQA initialises the derivative estimate with at least N+1 (1441) evaluations, but preferable uses 2N+1, and then builds an approximate Hessian which is updated throughout the optimisation. Even on simple convex surfaces, and even though BOBYQA is known to be very efficient, it will take many thousands of function evaluations to converge. Powell reports that a 320 dimension problem would take over 11,000 function evaluations.

Genetic algorithms have a large number of hyperparameters. Differential evolution, particle swarm and cross over operators tend to stay on the subspace hyperplane, whereas mutation operators can move away. Mutation is notoriously slow at moving around high dimensional spaces. Hence we would prefer a population size of at least 1440 in order to explore the space. Genetic algorithms are also very slow at convergence, with erratic behaviour, so stopping criteria should be very conservative. It is easy to go through thousands of function evaluations without any improvement, and then jump to an improvement.

Chen et al [5] report that for a simple problem of dimension 500, with the optimum at the origin, using particle swarm and differential evolution genetic algorithms, over a million function evaluations are required for convergence. For the Olympus challenge with 1440 variables in a complex space, we would expect many millions of function evaluations to be required. Genetic algorithms, in order to be successful, need to considerably reduce the search space dimensionality.

For proxy-based methods, until N+1 (1441) data points are available, the proxy is only reliable on the subspace hyperplane.

In contrast to optimisation problems, when using proxy models in history matching, far fewer data points are required as we are trying to match multiple history points. Here the guideline is to use roughly the same number of history match points as variables, so long as each history match point is giving new information. By following this guideline, history matches may emerge in less than 100 simulation runs even when there are over 1000 uncertain reservoir modelling parameters.

So, for a problem in 1440 dimensions, as a rough guideline, one might expect adjoint/quasi-Newton methods to require at least 2,000 evaluations, ensemble methods 70,000 function evaluations (50 iterations of a 1440 ensemble), and genetic algorithms well over a million function evaluations.

As a rough guide, proxy models may require a similar number of function evaluations to adjoint methods, say 2,000.

Our experience during the proxy-based expected improvement optimisation are:

* GA stage population size 400, 3500 function evaluations, function improves from 2.05 to 2.22
* Quasi-Newton BFGS stage 600 free variables, 340 function evaluations, function improves from 2.22 to 2.55

Clearly the GA stage has nowhere near converged. In contrast strict convergence criteria are used for the BFGS stage.

We would expect adjoint methods applied directly to the simulator to behave in a similar manner. So even if we have adjoints, it would still take at least 2000 simulations. Genetic algorithms would take many tens or hundreds of thousands of simulations.

These numbers need to be multiplied by 50 if we are doing optimisation under uncertainty, unless, as in our case, we can build a proxy model which captures the behaviour of all 50 geological models.

# NPV calculation

To calculate the npv, the EssRisk results calculator was used. This can calculate, based on a special scripting language, derived values for wells, groups and fields, using as input reservoir simulation results. It is also able to perform time period partitioning. The npv calculation is done for each time step, using a trapezoidal rule, so is slightly different to the calculation defined in the challenge, which is done on 3 month periods.

# Problem

The key to our approach is to use a large number of proxy models. The overall nvp value may be partitioned by well and time period:

Each partitioned npv value has a separate individual proxy model. In addition there is a proxy model for the overall npv for each well and one for the field. This means that there is a total of 1440 + 18+ 1 = 1459 proxy models.

We decided, for performance reasons, not to go to this extreme, but to partition only over time

so we have 80 + 1 proxy models (80 npv time periods, plus one for the overall npv). It is not necessary to have the extra one proxy model but is useful for checking and visualising the results.

For the optimization stage, the function evaluation involves a sum over geological models, so each function evaluation requires the calculation of 81 x 50 = 4050 proxy model estimates and derivatives. Each individual proxy model calculation requires the calculation of the distances to the existing simulation runs, which involves O(NrNm) calculation, where Nr is the number of runs and Nm is the number of modifiers.

These calculations only become feasible if we recognise that a proxy model for a particular time period is only dependent on modifiers for earlier times. In addition, the npv interest rate calculation means that later times are less influential on the overall npv value.

# Results

We show results from three cases. The first is to optimise a single geological model (number 25) without any uncertainty. This is shown in Fig. 1. We can see that there is little improvement beyond around 2000 simulation runs, which is in accord with our predictions.

The second study was to optimise under uncertainty using rate controls. This is shown in Fig. 2. There is, as expected, a more gradual improvement, and the study was stopped after 2700 simulation runs. We can see that the different geological models have stratified, with the low case being most apparent. There seems to be little scope for improving the low case under the uncertainty optimisation, but of course it could well be improved if optimised independently, as previously discussed. In addition, it is clear that the variation in objective between geological models is at least equivalent to variation within control parameters.

The third study was to optimise under BHP control. This can be seen in Fig. 3. The initial runs are the Latin Hypercube experimental design runs, and they have a strong tendency under BHP control to shut off amd have very poor npv. However, the results soon become superior to those obtained from rate control.

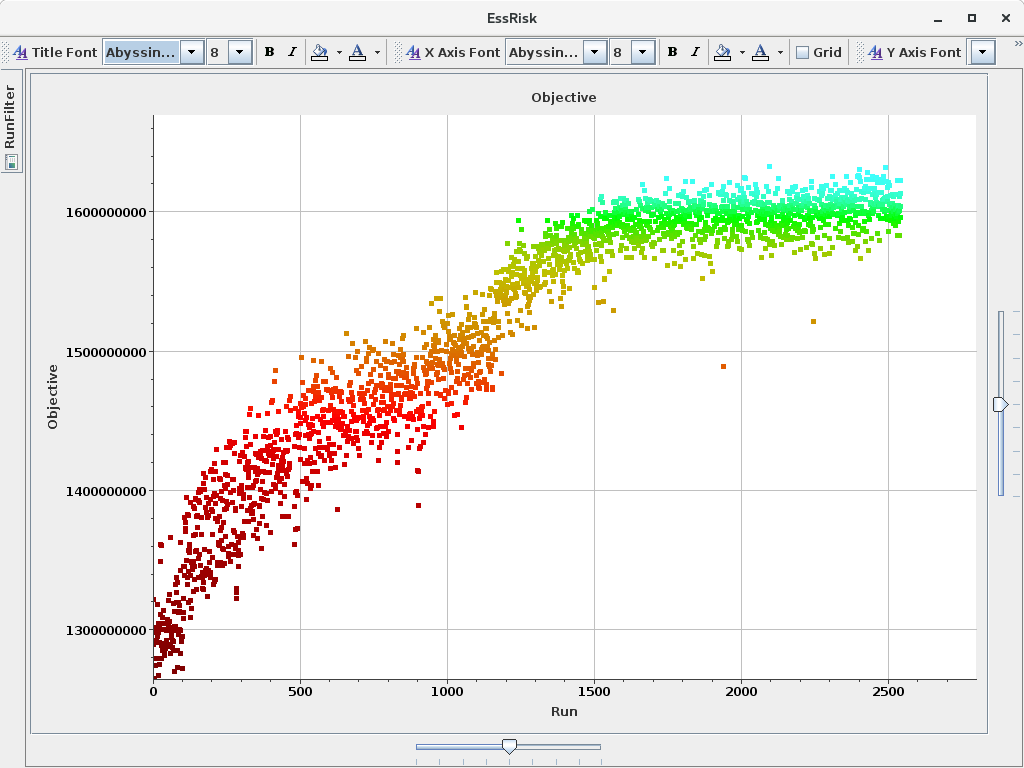


Fig. 1 Objective v. run for a single geological model

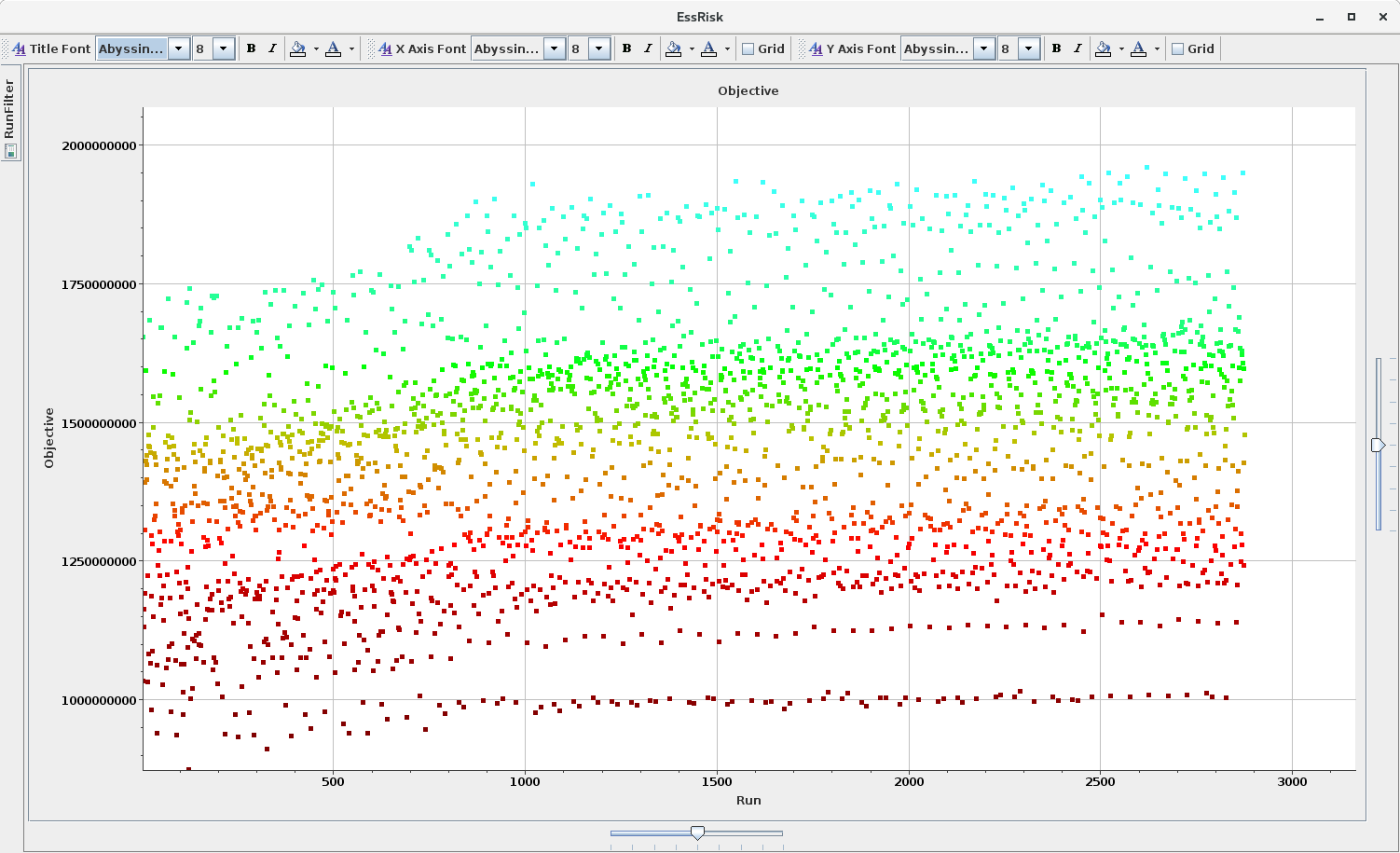


Fig. 2 Objective v. run under uncertainty with 50 geological models under rate control.

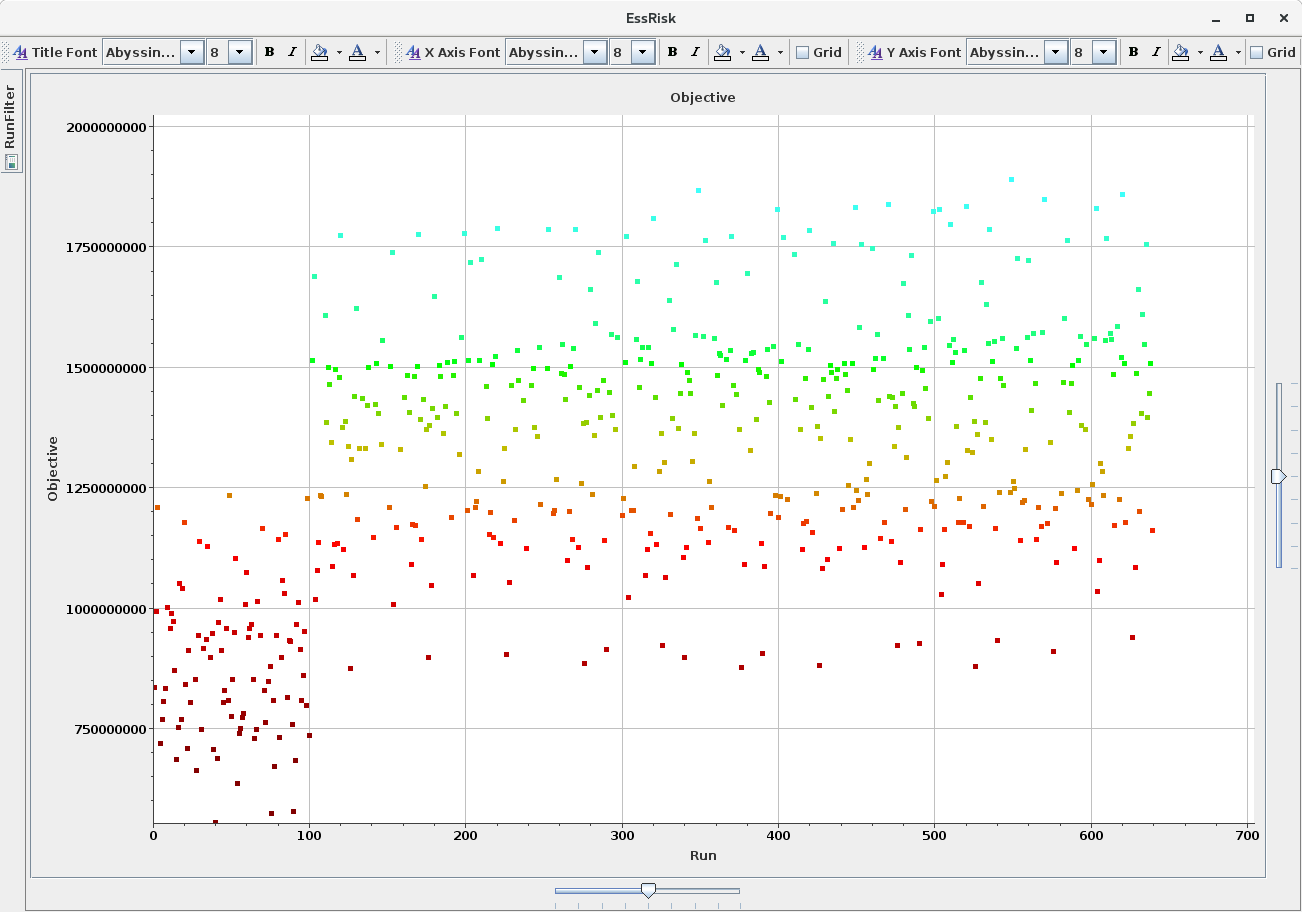


Fig. 3 Objective v. run under uncertainty with 50 geological models under BHP control.

Fig. 3

# Performance and hardware

The hardware platform for this work was a modest Linux workstation. The CPU was a single socket Intel i7-5280, 3.30 GHz with 12 GB memory with 6 cores, together with a single NVIDIA Titan Black GPU processor based on the same Kepler technology that powered Oak Ridge National Laboratory’s TITAN computer. It uses the GK110B chip, which has 2880 CUDA cores, 6GB of GDDR5 memory and has a higher double precision performance. The proxy model calculations take full advantage of the number of cores but did not use the GPU. All linear algebra was written in Java.

The reservoir simulator used was Echelon from SRT. This is specially written to take advantage of the GPU, and simulation runs took around 2-4 minutes.

The main bottleneck in the proxy model performance was the optimization of the expected improvement, as this involves iterations over both proxy models and geological models. Each optimization took around 5 minutes. This could be improved by adding cores – the software is written to be fully concurrent and scales linearly with the number of cores. There is also scope for improvement by using GPU’s with standard CUDA libraries.

# Conclusions

We have shown the potential of the use of proxy models for optimisation under uncertainty, without recourse to expensive hardware. Optimisation can be performed on standard desktop computers used by reservoir engineers. We took a somewhat extreme approach; greater computing power (such as a 4 socket Xeon platform with 64 cores) or adjustments to the algorithms (such as using sparse Gaussian processes or using a proxy model to emulate a batch of runs rather than a single run) would improve the results. There is also the potential of reducing the number of modifiers (say yearly in later years). However, we hope we have demonstrated the power of carefully implemented proxy models, and have demolished the myth that they cannot be used on large scale problems.

**References**

1. Goodwin, N. H. Bridging the Gap Between Deterministic and Probabilistic Uncertainty Quantification Using Advanced Proxy Based Methods. *SPE-173301* (2015).

2. Goodwin, N. H., Esler K., Ghasemi M., Mukundakrishnan, K, Wang, H., Gilman, J. R., Lee, B . Probabilistic Uncertainty Quantification of a Complex Field Using Advanced Proxy Based Methods and GPU-based Reservoir Simulation.. SPE 182637 (2017)

3. Gohari K, Jutila H, Mascagnini C, Gryaznov A, Goodwin N; Howell M, Kidd,P. J, Bijani B. Novel Workflow for the Development of a Flow Control Strategy with Consideration of Reservoir Uncertainties. SPE-177427 (2015).

4. Jutila, H. A., & Goodwin, N. Schedule Optimisation To Complement Assited History Matching and Prediction Under Uncertainty. Vienna, Austria: SPE 100253 (2006).

5. Chen, Stephen & Montgomery, James & Bolufé-Röhler, [Antonio](https://www.researchgate.net/profile/Antonio_Bolufe-Roehler?_sg=X1x_4_99nLx9GqnPessfpXVPUo0vUtQTmGnsAhgnSEVig3AgsTHXOkbL__l55pU-u0ntFzk.cY5Ne5g_MtatRU8i8aEUI4XbxBNxAjPBw48ntYbNW2CbtC3KP4vNhGqToObizkaVVtyahhlvkgO1rTkXNczsTg) Measuring the curse of dimensionality and its effects on particle swarm optimization and differential evolution. Applied Intelligence. (2015).

6. Powell, M. J.. The BOBYQA algorithm for bound constrained optimization without derivatives. *Cambridge NA Report NA2009/06, University of Cambridge, Cambridge*. (2009).

7. Byrd, R. H., Lu, P., Nocedal, J., & Zhu, C.. A limited memory algorithm for bound constrained optimization. *SIAM Journal on Scientific Computing*, 16.5 1190-1208. (1995).