SO methods

Graham Burgess

February 2024

1 Introduction

How we categorise simulation optimisation (SO) methods:

- decision variable discrete or continuous. In the discrete case:
  - Finite or infinite number of feasible solutions
  - Ordered or unordered feasible solutions
  - Random search used or not. For random search methods:
    - neighbourhood structure (adaptive usually better than fixed)
- how the problem may be constrained:
  - unconstrained, partially constrained, fully constrained
  - stochastic constraints, deterministic constraints, box constraints
- dimensionality of the problem
- local or global solutions found
- convergence guarentee

The problem we face in the housing waiting-list problem has the following corresponding characteristics:

- In reality the decision variables (numbers of units of housing/shelter to build) are discrete, however we may work with a continuous approximation of the system. In the discrete case:
  - We have a finite but large number of feasible solutions
  - Within each dimension of the solution space, solutions are ordered, however ordering across dimensions is not obvious
- The problem we currently have in mind is:
  - fully constrained
  - deterministic constraints

- There is potentially a medium number (approx. 10) of dimensions (how much of housing/shelter to build each year over some reasonable time horizon for planning

- We would like a global solution, however we may still be interested in locally convergent methods provide we can add an element of random searching to escape any local optimum if necessary.

2 Integer ordered decision variables

This occurs when the decision variables are numerical in nature and only take integer values (e.g. how many staff to hire, how many houses to build) - can be seen as the integer points on a lattice.

2.1 Discrete Stochastic Approximation

The idea is that the objective function on the discrete space (say f) is extended to the continuous space (say $\tilde{f}$) and SA is performed on this extension. Therefore at each iteration, the step to a new solution is not restricted to an integer point, which means it is 'free' to quickly move towards the optimal solution. Finally, one rounds to the nearest integer point.

In the 1-D case, a simple linear interpolation between points in f is performed to construct $\tilde{f}$. Then, following an initial guess θ_0 of the minimiser of $\tilde{f}$, gradient descent steps are taken using a gradient estimator obtained via (multiple replications of) simulation of the integer points either side of θ_0. The step sizes are a sequence of positive numbers.

When $d > 1$, the extension from f to $\tilde{f}$ for $d > 1$ is done by partitioning the continuous space (using a colletion of convex hulls built on points in the integer space) so that every point in the continuous space is contained within one of the convex hulls in the collection. Then $\tilde{f}$ is constructed as a linear combination of f at all points on the surrounding convex hull. The weights in the linear combination depend on the proximity of the point in continuous space, to the corresponding points on the convex hull. Gradients are constructed using estimates (via multiple replications of simulation) of f at the points on the convex hull. At each iteration, once a gradient is constructed, one moves to a new point by taking a step along the gradient, with the step sizes a sequence of positive numbers. Now that the dimensionality of the problem is greater than 1, multimodularity of the objective function (or L^{sharp}-convex) (as defined in the paper) is required to guarantee the convergence of this algorithm, sinc for multimodular (or L^{sharp}-convex) functions, the extenstion to the continuous space gives a convex function.

Step-size at iteration n is of the form $a/(b + n)$ where a and b are positive constants. b is chosen as the total number of iterations and a is selected by the user based on what is most advantagous for an initial step-size.

2.2 R-SPLINE

R-SPLINE is "retrospective" in that it solves a series of sample-path problems, using the previous solution (i.e. retrospective) as a starting point for each new sample-path problem. Each sample-path problem is generated implicitly (i.e. sample-paths are collected as needed as the algorithm is progresses). Each sample-path problem is solved using a repeating combination of a continuous line search, and a neighbourhood evaluation.

The line search moves along "phantom" gradients from integer point to integer point in search of a better solution. Each move starts with a random perturbation from the current best (integer) solution to a continuous point. Then the objective function value and gradient at this point is computed with a linear interpolation very similar to that in DSA. There is then "continuous" movement along this gradient and a jump to the nearest integer point, in search of an improved integer solution. The step-size at each continious movement is calculated to be as small as possible so that the jump to the nearest integer point will land on a new integer point. The line search stops when a move within the search is small (below some threshold). Once a line search is complete, a neighbourhood evaluation is performed to check that the current best is indeed a local solution (within all of its nearest neighbours, in every dimension). The combination of the line search and the neighbourhood evaluation is repeated until no improvement is found, or until the total simulation budget is used.

There is similarity between R-SPLINE and DSA in the way the objective function value and gradient in the continous space are calculated. However, there are differences: DSA, once transformed into the continous space, only moves in the continuous space, whereas R-SPLINE regularly jumps to the nearest integer point, before randomly perturbing away from that integer point. R-SPLINE involves repetitions of the line search and a neighbourhood evaluation, however DSA performs no neighbourhood evaluation as it moves towards the optimal solution. There is also a difference in how the step size is calculated as the search moves along the gradients. R-SPLINE has has an adaptive step-size (explained above), whereas DSA gets smaller and smaller as the algorithm progresses. Both algorithms increase the number of simulation replications from iteration to iteration.

More recently, a new algorithm called "ADALINE" incorporates many features of R-SPLINE but invovles an adaptive sample size at each iteration so as not to waste computation budget.

2.3 COMPASS

Convergent optimisation via most-promising-area stochastic search (COMPASS). Based on random search (like many other DOvS algs in literature) but offers a unique adaptive neighbourhood structure known as the "Most promising area" (MPA). For terminating and steady state simulation. For fully/partially/un-constrained problems. The fully-constrained setting is described as follows:

The algorithm starts by simulating an initial solution and the entire feasible region is in the MPA. Then, new solutions are sampled from the MPA and simulated. Once simulation has taken place, the "promising"-index of each feasible solution is the sample mean performance of the visited solution which is closest to it. Therefore, the MPA is the set of feasible sols which have the current sample best solution as their closest visited solution. By design, the only previously-visted solution in the MPA is the current best. At each iteration, COMPASS updates the current best solution and the MPA and assigns simulation experiments (using a Simulation Allocation Rule) to candidate solutions sampled uniformly from the the MPA. It keeps information from previously-visited solutions over the course of the algorithm.

When the problem is partially- or un-constrained, a boundary around the current best solution is maintained and is widened as candidate solutions get close to the boundary. Within the boundary, the algorithm works as in the above fully-constrained description.

If the algorithm gets stuck in the wrong area (due to simulation noise) because all visited solutions are allocated further simulation budget, the algorithm will eventually move away from the sub-optimal area. The user must think carefully about how many solutions to sample from the MPA at each iteration. Fewer will reduce the time taken to find a local optimum, but the quality of this local optimum may not be good.

2.4 GMRF

Model execution is too slow to model even a fraction of the optimal solutions. Ranking and selection typically simulate all and then reach some guarantee of PCS (frequentist). Adaptive Random Search if its impossible to simulate everything. this paper looks for R+S style guarantees with an efficient search, learning the spatial correlations along the way. Multi-resolution framework: region-level learning and solution-level learning.

The precision matrix is the inverse of the covariance matrix for a random vector. The diagonals are 'conditional precisions' - i.e. the inverse of the conditional variances (the variances given knowledge of all other elements of the random vector). The non-diagonal elements are 'conditional covariance' - i.e. the covariance given knowledge of all other elements of random vector and can be normalised to 'conditional correlation'. The proof of this can be found in the textbook GMRFs by Rue and Held - it stems from definitions of the conditional distribution (i.e. proportional to the joint distribution) and the definition of conditional independence (i.e. $\pi(x_i, x_j, x_{\{-ij\}}) \propto f(x_i, x_{\{-ij\}})g(x_j, x_{\{-ij\}})$) which holds when $Q_{ij} = 0$.

There is some relevant discussion of matrix algebra concepts such as symmetric positive definite matricies and choleskey decompositions in the book by Rue and Held. There are also helpful relations involving the conditional distribution $\pi(x_A|x_B)$. With fairly basic matrix algebra (and again starting with the definition of the joint distribution, one can derive expressions for

$\mathbb{E}(x_i|x_{-i}), \mathrm{Prec}(x_i|x_{-i})$ and $\mathrm{Corr}(x_i, x_j|x_{-ij})$. The latter being: $\mathrm{Corr}(x_i, x_j|x_{-ij}) = -Q_{ij}/(Q_{ii}Q_{jj})^{1/2}$.

The neighbourhood structure of a graph can define the non-zero entries of a precision matrix. The authors introduce parameters $\boldsymbol{\theta}$ where θ_0 is the conditional precision of each solution and θ_j is the conditional correlation if two solutions differ by one unit in the jth dimension. This means response surfaces can change more rapidly in one direction compared to another. $\boldsymbol{\theta}$ entries are all between zero and one, we want positive correlations but should be below one. These parameters are estimated with MLE and the initial simulation output, using profile log likelihood.

So random vector Y is unknown, $\boldsymbol{y} = \mathbb{E}[Y]$ is true objective function. $\boldsymbol{y}$ is also unknown but we model it as $\mathbb{Y}$, a realisation of a GMRF as defined. What we observe is y + noise which is modelled as another GMRF: $\mathbb{Y}^{\epsilon} = \mathbb{Y} + \epsilon$. The noise is modelled as Gaussian with diagonal precision matrix if CRN not used. Interested in the conditional distribution of $\mathbb{Y}$ given the simulation output at the design points (feasible solutions which have been simulated). Given a vector of sample means at the design points, the authors prove a conditional distribution of $\mathbb{Y}$. They do this by first writing out the joint distribution of $(y_1, y_2, y_2^{\epsilon})$ where subscript 2 and 1 represent solutions which have and have not been simulated, respectively.

Optimisation performed using Complete Expected Improvement which averages over the both full conditional joint distribution of $\mathbb{Y}$ and the simulation noise. EI estimates the optimality gap given the current data (Knowledge gradient looks at the predictive distribution from simulating other solutions more and picks that which predicts the best improvement - useful when more simulation is expensive and you want to see if its worth it. The notion of EI comes from Jones et al and the idea is possible improvements (since the objective value at different points is unknown) are weighted by the prob density at these opints. In original EI for deterministic models, the difference between current best (with known value) and new point was modelled as normal rv, in our problem the current best is also rv, but the difference is still a normal rv. The closed form expression for EI comes from translating the expected vallue into an integration and doing integration by parts.

SKO by Huang 2006 also discusses efficient black box estimation with sequential kriging, and their second paper in 2006 discusses a multi fidelity version where they introduce augmented EI which considers the fidelity level of the proposed point using α_1 (correlation between two fidelities - accounts for reduction in reward if lower fidelity model used), α_2 (accounts for diminishing returns for extra replicates as prediction becomes more accurate with model of proposed fidelity) α_3 (the ratio of the cost between the proposed system and the hightest fidelity system).

Question: Claim is that the average is over the full conditional joint dist of $\mathbb{Y}$ and the simulation noise, but isn't the latter incorporated into the former in the conditional distribution of the former? Is the simulation noise maybe also explicitly acknowledged because the CEI takes a difference between two random variables (obj val at current best and add proposed)?

The solution-level algorithm is what I'd expect, except interesting to note that after the candidate solution is found which maximises the CEI, extra simulation budget is expended on *both* that solution *and* the current best, before proceeding.

At each computation of the CEI (note: this computation is needed for n-1 solutions at every iteration of the algorithm) - the conditional distribution $\mathbb{Y}$ given the data is estimated, using estimates of parameters within the matrix. The computationally expensive bit is inverting the conditional precision matrix. But because it is sparse (due to neighbourhood structre and Markov property and due to the diagonal nature of the intrinsic noise matrix) and because only a small number of matrix elements are actually needed for the computation, matrix techniques which exploit these characteristics can be used.

The specifics of the sparse-matrix techniques are not detailed in this paper. But reference is made to the PARSIDO project, and the authors also say that 'general-purpose' sparse matrix methods are used for the purpose of giving a 'proof of concept' of the GMRF.

Computing the CEI:

- Original GMIA paper: not clear exactly, but nothing very smart - 'general purpose matlab methods' - papers which cite this paper say they do the $diag(\bar{Q}^{-1})$ with full backsolve of $\bar{Q}\bar{Q}^{-1} = 1$ (which involves decomposition, then forward and backward substition, where you have to solve for a nxn matrix with this method) and the specific column of $\bar{Q}^{-1}$ using a simpler backsolve (i.e. you solving for an nx1 column instead. M^t is also cheap because it is solving for a nx1 vector not a nxn matrix.
- Takahashi: a way of computing a diagonal element without needing to compute the full inverse, but still lots of computations for large solution space.
- Semelhago 2017: trick of only periodically computing $V^t(x)$, and estimating subsequent $M^{t+1}(x)$ using trick with Sherman-Morrison-Woodbury identity to make it cheap as only a small number of elements of $\hat{Q}$ change from iteration to iteration. Again exploiting that updating M is cheap. They also have the idea of selecting top q solutions at each iteration for further simulation.

The convergence of this algorithm to the optimal solution as the number of sim reps goes to inf is based on the proof that with probability one, every solution will be simulated infinitely often given this algorithm.

A regional level approachis also given where there are GMRFs where each node corresponds to a region of the solution space, and there are also within-region solution-level GMRFs. This type of approaches is considered helpful for a problem with low dimensionality but a very large number of solutions.

2.4.1 Extenstions to GMRF

There are several new algorithms which extend GMIA in one way or another. Examples include projected GMIA (pGMIA) (Li and Song), rapid GMIA (rGIA) (Mark Semelhago) and additive GMIA (aGMIA) (Harun) - these mostly address computational challenges e.g. in matrix computation.

*Computational Methods for GMRFS (Semelhago et al 2017 at WSC) - first up shows how the the backsolve method words using a LDL facorisation to compute the $C(x, \tilde{x})$ column of $\bar{Q}^{-1}$. The hard bit is then computing the diagonal of $\bar{Q}^{-1}$. This can also be tackled using a LDL factorisation: $\bar{Q}\bar{Q}^{-1} = 1$, $\bar{Q} = LDL^T$, and this method is the approach of the original GMRF paper. Takahashi show a neat way of calculating specific diagonal elements of $\bar{Q}^{-1}$ with few calculations (Example of this given in this Semelhago paper.) Semelhago have approaches for improving the Takahashi approach still. It seems that Cholesky factorisation is expensive but it is a similar approach to LDL.

Projected GMIA: The computational cost of inferences increases exponentially with number of dimensions in GMIA. pGMIA (and pGMIA+) seems similar to the multi-resolution GMIA proposed in the original GMRF paper, however pGMIA exploits the depdence between layers since there is dependence due to averaging. But basically it collapses dimensions by taking an average. It then does a region layer search and a solution layer search when deciding where to sample next. It bascailly 'projects' the solution space onto a lower dimensional space.

Rapid GMIA: Infrequent evaluations of full conditional dist with rapid learning on a smaller subset of solution space. Solutions in the subset need not be close to each other (key to the efficacy of the alg) Effectively at each iteration, a subset of promising solutions (ranked by sample means) are selected and only the conditional distributions for these solutions is updated while searching for a good next solution to evaluate. The promising set would typically include solutions near the current best, solutions with promissing sample means and solutions in unexplored regions - this set would typically thne include solutsion from across the solution space - keeping it global.

Fewer diagonal elements of $\bar{Q}^{-1}$ are required, and no expensive Cholesky factorisation needed in rapid search steps - only a simple linear alg step.

The concept of a promising set is put together at the end of each gloabl search - i.e. the top n solutions ordered by their CEIs. Then the the $\bar{Q}$ matrix can be partitioned and the rapid search iteratively done only needing the small partition of this matrix relating to the search set. manipulation of dense Σ is favoured over Q here. There is then a stopping criterion for the rapid search - the authors propose stopping the rapid search when all of the CEIs for solutions in the search set are worse than the solution with the best CEI in the fixed set. The authors highlight several of the steps which are the most computationally expensive and specify how they tackle these expensive bits and how their partition leads to a significant saving compared to GMIA. I suppose there must be a (statistical) risk that when you collect new data in the search set but don't update the CEIs in the fixed set, the next solution you choose (based on CEIs in the search set) hasn't got the best global CEI. Can't see exactly if/where the authors address this issue.

In the paper, i am STUCK at the point where it says 'computing the column of $\bar{Q}^{-1}$ corresponding to $\tilde{x}$ requires solving the system $\bar{Q}z = e_{\tilde{x}}$ for z. This apparently requires factorising $\bar{Q}$. Also I am STUCK with Lemma 2.1 in Rue and Held which proves the conditional distribution of the main GMRF paper.

Additive GMIA Computational cost of posterior update increases quadratically with the number of feasible solutions. Decompose the prior into additive form as an approx.

iGMRFs. Problems with original parameterisation of the precision matrix and estimating the parameterrs wiht MLE:

- positive-definiteness must be checked for every allowable combination of param vals
- small param values lead to a mean-reverting GMRF. (see paper for nice example of this)
- further more: interepretation of θs is different for feasible points in the interior and on the boundary of the feasible region. (again, see paper for nice example)

First order iGMRFs are suitable when we have prior belief about the structure of the precision but do not have a prior belief about the mean. They are invariant to the addition of a constant.

2.4.2 Multi Fidelity GMRF

There is also a paper on using multi fidelity models within the GMRF framework. Rough outline of the features of this approach:

Assume no heirarchy in the models of different fidelity. Assume computational cost is negligible of low fidelity models. Assume already know the low fidelity evaluations for entire solution space.

- Graph duplicated for each low fidelity model used, to represent the low fidelity responses. Each 'layer' connected to high fidelity 'layer' with an edge (i.e. conditional correlation) but no edges between solutions in a layer

- Different constant μ_i used as the 'initial' prior information for each model i (each 'layer' in the graph), but the constant still appears to be the same across each solution for a model. This constant 'estimated' using the initial simulation output from a selection of solutions - not clear how this is estimated. Perhaps this prior can be bet
- Additional parameters (ρ) (which must be estimated using MLE) to indicate the conditional correlation of each solution so its counterpart in the other low fidelity models. This is the means by which information from the low fidelity can influence the search of the algorithm.
- The precision matrix at each iteration of the routine is updated to incorporate the additional 'precision' which is brought about from the extra high-fidelity simulation which has been performed. This effectively reduces the conditional correlation between a solution and its low-fidelity coutnerparts, and increases the conditional correlation between solution neighbours on the high-fidelity 'layer'.
- The authors then have a theorem (without proof) which gives the conditional distribution for the GMRF at iteration t, which is equivalent to the conditional distribution given in the origianl GMRF paper but there is an additional term which accounts for the additional information from the low fidelity model(s). It is supposed that this term goes to zero as more and more high fidelity simulations are performed.
- Their algorithm is what you might expect. They have to update the precision matrix at each iteration because it incorporates the changing conditional precisions as more simulation is performed.
- They compare their algorithm to GMIA and also to a version where the model the error between high and low fidelities with a GMRF.

My overall thoughts are that they have some good ideas but I think there could be better ways of incorporating the information from lower fidelity models most efficiently. For example, they still have flat priors for each model. They also have to recalculate the precision matrix at each iteration which presumably is costly.

2.4.3 GIBBON

Information theortic BO seeks to reduce the uncertainty in the location of good areas of the search space, as measured by differential entropy. MAx-value entropy search (MES) reduces the uncertainty in the max value of the objecitve func, wheras other acquisition funcs (ES, PES) reduce uncert in location which is only possible in Euclidean search spaces and also requires expensive approximation schemes. But using MES in common BO extensions (like Multi fidelity BO - MUMBO) requires further approximations which are expensive.

Shannon's entropy for discrete prob dists is $-\sum_x p_x log(p_x)$. Low prob things have small p_x and extremely negative $log(p_x)$ and highly prob things have bigger p_x and still negative but less negative $log(p_x)$. it turns out that the more spread out the distribution is, the higher this value, and the more precise the distribution is, the smaller Shannon's entropy. Differential entropy is the same on the continuous domain.

2.5 RSM in a discrete setting

Response Surface Optimization with Discrete Variables (2004)

3 Continuous decision variables

3.1 Sample Average Approximation

"A Guide to Sample-Average Approximation" - Kim, Ragu, Shane Henderson Multi-dimensional newsvendor problem analysis shows that differentiability and expectation are interchangeable. The true function and the sample problem have the same nice properties of smoothness and concavity. So then - sufficiently large N -¿ effectively concave and deterministic opt can be used to find optimum.

We can say that SAA is appropriate for a problem if there is a structure to the sample average function which allows a nice deterministic optimisation solver to be used, and that the true function can be said to share that same structure.

3.1.1 Infinite dimension SO problems - (function decisions) - approx with SAA

Singham et al.

3.1.2 Retrospective framework

How this differs from SAA

3.2 Stochastic Approximation

3.3 Meta-modelling

3.3.1 Trust-region methods

3.3.2 Response Surface Methodology

STRONG

3.3.3 Global meta models

BO

3.3.4 Multi-fidelity models

A simulation-based optimization algorithm for dynamic large-scale urban transportation problems - Osorio and Chong 2018

Zirui Cao et al. CLUSTER-BASED SAMPLING ALLOCATION FOR MULTI-FIDELITY SIMULATION OPTIMIZATION (WSC 2023)