The goal of this method is to evenly sample the near-optimal feasible decision space of a linear optimization model using a version of Markov Chain Monte Carlo sampling named the hit and run algorithm.

1 Model

The model under consideration is linear and convex and can be written on the standard form:

$$\begin{aligned} \text{minimize} \quad & \mathbf{f}_0(\mathbf{x}) && (1)\\ \text{subject to} \quad & \mathbf{f}_i(\mathbf{x}) \leq 0 \;\; i = 1...m && (2)\\ & \mathbf{h}_j(\mathbf{x}) = 0 \;\; j = 1...p && (3) \end{aligned}$$

With $\mathbf{x} \in \mathbf{R}^n$. Besides from the constraints contained in the original optimization problem, an additional constraint is introduced, to separate the near-optimal feasible space, from the entire feasible space.

$$f_0(\mathbf{x}) \leq f_0(\hat{\mathbf{x}}) \cdot (1 + \epsilon) \tag{4}$$

Where $\hat{\mathbf{x}}$ is an optimal solution to the original problem and $f_0(\hat{\mathbf{x}})$ is the objective value for the optimal solution. The slack on optimality is given by the slack variable θ .

As the constraints are linear they can be written as matrix products on the form:

$$\begin{aligned} \mathbf{f}_i(\mathbf{x}) \leq 0 \Rightarrow A\mathbf{x} \leq \mathbf{b} && (5)\\ \mathbf{h}_j(\mathbf{x}) = 0 \Rightarrow H\mathbf{x} = \mathbf{c} && (6) \end{aligned}$$

The set containing all feasible near-optimal solutions then become:

$$F = \{\mathbf{x} \in \mathbf{R}^n | A\mathbf{x} \leq \mathbf{b} \wedge H\mathbf{x} = \mathbf{c}\} \tag{7}$$

2 Presolve

In order to apply the hit and run sampling algorithm, a fully dimensional space is required. As several equalities are included in the definition of F and hidden equalities might be found among the inequality constraints, the fully dimensional subspace of F must be found. Hidden equalities occur when two inequalities limit the range of a variable to zero.

The goal of the presolve process is to define the fully dimensional subspace Z of F.

Initially all linearly dependent constraints/rows in the augmented matrix $[A|\mathbf{b}]$ are identified. Linearly dependent constraints/rows, span parallel

hyper-planes in F. If two such hyper-planes were to coincide, and have normal vectors pointing in opposite directions, they constrain a dimension of F and can be represented as an equality rather than two inequalities. For all such inequality constraints constraining a dimension of F, their given row in A are moved from the matrix A to the H matrix.

From [1] (p. 523) we know that the problem can be reformulated as:

$$\begin{aligned} \text{minimize} \quad & \mathbf{f}_0(\hat{\mathbf{x}} + N\mathbf{z}) && (8) \\ \text{subject to} \quad & A(\hat{\mathbf{x}} + N\mathbf{z}) \leq \mathbf{b} \quad i = 1...m && (9) \\ & span(N) = \mathcal{N}(H) && (10) \end{aligned}$$

Where the span of N is the null space of H and $\hat{\mathbf{x}}$ is any particular solution contained in F. As all equalities have been eliminated from the problem, the fully dimensional subspace of F can be defined as:

$$Z = \{\mathbf{z} \in \mathbf{R}^{n-p} | A(\hat{\mathbf{x}} + N\mathbf{z}) <= \mathbf{b}\} \tag{11}$$

Using this the optimization problem can be reformulated as:

$$\begin{aligned} \text{minimize} \quad & \mathbf{f}_0(\hat{\mathbf{x}} + N\mathbf{z}) && (12) \\ \text{subject to} \quad & \hat{A}\mathbf{z} \leq \hat{\mathbf{b}} && (13) \end{aligned}$$

With $\hat{A}$ and $\hat{\mathbf{b}}$:

$$\begin{aligned} \hat{A} &= A \cdot N && (14) \\ \hat{\mathbf{b}} &= \mathbf{b} - A\hat{\mathbf{x}} && (15) \end{aligned}$$

By validating that the matrix $\hat{A}$ has full rank, we know that the subspace Z is fully dimensional.

3 Hit and run sampling

A version of the hit and run algorithm [2], is used to sample the subspace Z. The algorithm consist of an iterative process of drawing random directions, and taking steps of random lengths within the space Z.

Initially a point $\mathbf{z}_0$ inside the space Z must be provided. Random steps inside Z are then taken by generating a random direction θ and taking a random step t that will not violate the boundaries of Z.

$$\mathbf{z}_{i+1} = \mathbf{z}_i + \theta t \tag{16}$$

θ must be a unit vector pointing in a random direction evenly distributed on a unit hyper sphere $\mathbf{S}^{n-p}$. I will elaborate on how to do so!!! But this is easy.

To determine the range of t that ensures that the step t does not cross the boundary of Z, Equation 16 is substituded in to Equation 13.

$$\hat{A}(\mathbf{z}_i + \theta t) \leq \hat{b} \tag{17}$$

Isolating t will result in the upper and lower bounds:

$$t_{max} = \inf \frac{\hat{b} - \hat{A}\mathbf{z}_i}{\hat{A}\theta} \text{ given } \hat{A}\theta > 0 \tag{18}$$

$$t_{min} = \sup \frac{\hat{b} - \hat{A}\mathbf{z}_i}{\hat{A}\theta} \text{ given } \hat{A}\theta < 0 \tag{19}$$

Selecting t to be:

$$t = (t_{max} - t_{min})r + t_{min} \tag{20}$$

Where r is a random number drawn from a uniform distribution between 0-1.

Repeating the process of generating random directions θ and taking random step lengths t, will, if enough samples are drawn, generate a uniform sampling of the near optimal feasible set Z. Storing all samples z_i in the discrete set $Z^* = z_i \forall i = 0..d$ where d is the number of samples.

"The reason that the Markov chain corresponding to the iterates $Z_0, Z_1, ..., Z_d$ converges in distribution to a uniform distribution over Z is easily seen from the fact that 1) it is possible to go from any point in Z to a neighborhood of any other point in one step, and 2) the uniform distribution is a stationary distribution of the chain." [3]

The hit and run algorithm is md hard [4]. Where m is the number of constraints and d is the number of variables.

4 Decrush

Having sampled Z, these sample points must be reverted back to the F domain. This is done with a so called decrush algorithm.

$$\mathbf{x}_i = N\mathbf{z}_i + \hat{\mathbf{x}} \tag{21}$$

Repeating this for all samples a set containing d samples in F is obtained.

5 Further work

Figure out if sampling Z evenly, generates evenly distributed samples in F when samples are maped from Z to F.

As of now, i have had sucess with applying the method on problems with $n \approx 10.000$, and i believe that $n \approx 100.000$ is doable on a better pc. This is however one to two order of magnitude from the 1.000.000+ variables included in most modern energy system optimization models. Therefore i have considered two options. 1) Using decomposition methods, such as Benders decomposition to decompose the problem in to manageable sub-problems. Specifically it is only the non-timedependable variables that are truly of interest, and as these only make up ≈ 1000 of the variables in the energy system model. By decomposing the timedependable variables from the non-timedependable variables, and sampling the non-timedependable subspace, could be a solution. I have, however, not managed to figure out if this is a viable solution, as decomposition methods often require several iterations between the two sub-problems. 2) The largest contributor to the large number of variables in energy system models, is the time-series data used. By generating augmented time-series, using techniques such as self organizing maps [5], one can generate shorter time-series, thereby reducing the problem. This is however, a less favorable option than solution (1) as using augmented time-series

References

[1] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[2] Robert L. Smith. "Efficient Monte Carlo Procedures for Generating Points Uniformly Distributed over Bounded Regions". In: *Operations Research* 32.6 (1984), pp. 1296–1308. DOI: 10.1287/opre.32.6.1296. eprint: https://doi.org/10.1287/opre.32.6.1296. URL: https://doi.org/10.1287/opre.32.6.1296.

[3] R. L. Smith. "The hit-and-run sampler: a globally reaching markov chain sampler for generating arbitrary multivariate distributions". In: *Proceedings Winter Simulation Conference*. 1996, pp. 260–264.

[4] Claude Belisle, Arnon Boneh, and Richard Caron. "Convergence properties of Hit-and-Run samplers". In: *Communications in Statistics. Stochastic Models* 14 (Jan. 1998). DOI: 10.1080/15326349808807500.

[5] Hasan Umitcan Yilmaz et al. "Reducing energy time series for energy system models via self-organizing maps". In: *it - Information Technology* 61.2-3 (1Apr. 2019), pp. 125 –133. DOI: https://doi.org/10.1515/itit-2019-0025. URL: https://www.degruyter.com/view/journals/itit/61/2-3/article-p125.xml.