Risk Aware Economic Dispatch

Karan Chadha and Rahul Jain

December 14, 2017

Contents

1 Stochastic Programming

A good introduction to stochastic programming has been given in [6] and a book-length treatment has been given in [7]. Stochastic programming is a way to model uncertainty in optimization parameters. We will focus on two stage stochastic linear programs where the general problem can be defined as

$$\min_{x \in X} \mathbb{E}[f(x, \omega)] = \min_{x \in X} c^T x + \mathbb{E}[Q(x, \xi(\omega))]$$

which can be divided into two stages with the cost function of the first stage being

$$f(x, \omega) = c^T x + Q(x, \xi(\omega))$$

The second stage captures all the uncertainty and it occurs after the realization of the random variables. It can be described as the following optimization problem:

$$Q(x, \xi^s) = \min_{y^s} \{(q^s)^T y^s : T^s X + W^s y^s = h^s, y^s \geq 0\}$$

Here, s denotes the particular realization of the random variables. We notice that here both the constraints and the objective function may depend on the random variable and hence, may be random themselves. ξ captures all the randomness of the problem and can be considered as the tuple (q^s, T^s, W^s, h^s). Each tuple represents a scenario. Here, x and y are the first and second stage decision variables, respectively. Solving such problems exactly, is generally possible when the randomness is over a discrete space and only a finite number of scenarios or second stage problems are possible. The expectation in the first stage can then be calculated as the weighted sum of the second stage value function. This whole problem can be represented as a single linear program with a large number of variables and constraints as follows:

$$\begin{aligned} \min_{x \in X, y^s \forall s \in S} \quad & c^T x + \sum_{s \in S} (q^s)^T y^s \\ \text{subject to} \quad & T^s X + W^s y^s = h^s, y^s \geq 0, \quad \forall s \in S \end{aligned}$$

where S represents the set of scenarios.

But, when the number of scenarios increases it may not be feasible to solve the second stage problem for all the scenarios. Also, in the case of randomness over a continuous state space, exact ennumeration of all scenarios is not possible. In such cases, we are happy with approximate solutions with relatively lower computational effort. Such methods use standard sampling techniques such as Monte Carlo. Some of them have been described below:

The most direct sampling approach to the two-stage stochastic program is to replace the recourse function,$\phi(x)$, by a Monte Carlo estimate.

$$\phi(x) = \sum_{k=1}^{\nu} \frac{V(x, \xi_k)}{\nu}$$

1.1 L-shaped Method

In this method we solve the problem of the form:

$$\begin{array}{lll} \min & c^T x + \sum_{k=1}^{K} p_k q_k y_k & \\ \text{subject to} & Ax = b & \\ & T_k x + W y_k = h_k, & k = 1, \ldots, K \\ & x \geq 0, y_k \geq 0, & k = 1, \ldots, K \end{array}$$

That is, it gives exact solution to methods in which the random variable has a finite support.
Algorithm:
Step 0. Set $r = s = v = 0$.

Step 1. Set $v = v + 1$. Solve the following linear program

$$\begin{array}{lll} \min & c^T x + \theta & \\ \text{subject to} & Ax = b & \\ & D_l x \geq d_l, \quad l = 1, \ldots, r & (L.1) \\ & E_l x + \theta \geq e_l, \quad l = 1, \ldots, s & (L.2) \\ & x \geq 0, \theta \in \mathbb{R} & \end{array}$$

Let (x_v, θ_v) be an optimal solution. If no constraint (L.2) is present, θ_v is set equal to $-\infty$ and is not considered in the computation of x_v.

Step 2. Check if $x \in K_2$(the feasibility region for the recourse problem). If not, add at least one cut (L.1) and return to Step 1.
To check if $x \in K_2$, we can do the following: Solve the following for $k = 1, \ldots, K$

$$\begin{array}{lll} \min & w = 1^T v^+ + 1^T v^- & \\ \text{subject to} & Wy + Iv^+ - Iv^- = h_k - T_k x^v & (1) \\ & y, v^+, v^- \geq 0. & \end{array}$$

until, for some k , the optimal value w'$>$ 0.In this case, let σ_v be the associated simplex multipliers and define

$$D_{r+1} = (\sigma_v)^T T_k$$

and

$$d_{r+1} = (\sigma_v)^T h_k$$

to generate a constraint (called a feasibility cut) of type (1.3). Set r = r+1 , add to the constraint set (1.3), and return to Step 1. If for all k , w' = 0 , go to Step 3.

Step 3 For $k = 1, \ldots, K$ solve the linear program

$$\begin{array}{lll} \min & w = q_k^T y_k & \\ \text{subject to} & Wy = h_k - T_k x^v & (2) \\ & y \geq 0. & \end{array}$$

Let π_k^v be the simplex multipliers associated with the optimal solution of Problem k of the above type. Define

$$E_{s+1} = \sum_{k=1}^{K} p_k \cdot (\pi_k^v)^T T_k$$

and

$$e_{s+1} = \sum_{k=1}^{K} p_k \cdot (\pi_k^v)^T h_k$$

Let $w^v = e_{s+1} - E_{s+1}x^v$. If $\theta_v \geq w^v$, stop; x_v is an optimal solution. Otherwise, set s = s+1 , add to the constraint set (L.2), and return to Step 1.

Essentially, in the first iteration we solve the problem by neglecting some constraints,i.e. all the constraints of the type $T_k x + W y_k = h_k$. This lets us find the solution faster, then we check if the solution is feasible for the second order, if it is not, we add some constraints(in step 2) to make it feasible. Such constraints (L.1) are called feasibility cuts. We do this by solving problem (4), because if the optimal value is greater than zero, the constraints of our original problem are not satisfied. They are satisfied only when the optimal value to (4) is 0. If the solution is feasible, or made feasible by adding feasibility cuts, we move on to the next step to add optimality cuts which are linear approximations to ϕ on its domain of finiteness. In the case of piecewise linear ϕ, they are the support functions. Essentially, $e_{s+1} - E_{s+1}x$ is a support function of ϕ if it is piecewise linear, else it represents the tangent at the point x^v. Hence, for non-linear problems, here we find an envelope of piecewise linear functions which is a lower bound for the actual non-linear function and find its minimizer. When the probability space is not discrete, we can use Monte Carlo Sampling and replace all p_k by $\frac{1}{N}$ where N is the number of scenarios. This will give us few supporting hyperplanes instead of all and is a good approximation to the original problem.

1.2 Multi-cut Version

Step 0. Set $r = s_k = v = 0$ for all $k = 1, \ldots, K$.

Step 1. Set $v = v + 1$. Solve the following linear program

$$\begin{array}{lll} \min & c^T x + \sum_{k=1} K\theta_k & \\ \text{subject to} & Ax = b & \\ & D_l x \geq d_l, \quad l = 1, \ldots, r & (M.1) \\ & E_{l(k)} x + \theta \geq e_{l(k)}, \quad l(k) = 1, \ldots, s_k & (M.2) \\ & x \geq 0, \theta \in \mathbb{R} & \end{array}$$

Let $(x_v, \theta_1^v, \ldots, \theta_K^v)$ be an optimal solution. If no constraint (L.2) is present, θ_v is set equal to $-\infty$ and is not considered in the computation of x_v.

Step 2. As before

Step 3. For $k = 1, \ldots, K$, solve the linear program (5). Let π_k^v be the simpleax multipliers associated with the optimal solution of problem k. If

$$\theta_k^v < p_k(\pi_k^v)^T(h_k - T_k x^v)$$

define

$$E_{s+1} =]p_k \cdot (\pi_k^v)^T T_k$$

and

$$e_{s+1} = p_k \cdot (\pi_k^v)^T h_k$$

and set $s_k = s_k + 1$.If the above inequality does not hold for any k, stop. x^v is an optimal solution.

By adding disaggregate cuts, more detailed information is given to the first stage. The number of major iterations is expected then to be fewer than in the single cut method. In general, however, as numerical experiments reveal, as the number of major iterations is reduced, it is done at the expense of a larger first-stage program, because many more cuts are added. The balance between fewer major iterations but larger first-stage programs is problem-dependent. As a rule of thumb, the multicut approach is expected to be more effective when the number of realizations K is not significantly larger than the number of first-stage constraints m1.

1.3 Regularized Decomposition

It is a method that combines a multicut approach for the representation of the second-stage value function with the inclusion in the objective of a quadratic regularizing term. This additional term is included to avoid two classical drawbacks of the cutting plane methods. One is that initial iterations are often inefficient. The other is that iterations may become degenerate at the end of the process.

Algorithm:

Step 0. Set $r = v = 0, s_k = 0$ for all $k = 1, \ldots, K$. Select a^1 , a feasible solution.

Step 1. Set $v = v + 1$. Solve the regularized master program

$$\begin{array}{lll} \min & c^T x + \sum_{k=1} K\theta_k + \frac{1}{2}||x - a^v||^2 & \\ \text{subject to} & Ax = b & \\ & D_l x \geq d_l, & l = 1, \ldots, r \qquad (M.1) \\ & E_{l(k)} x + \theta \geq e_{l(k)}, & l(k) = 1, \ldots, s_k (M.2) \\ & x \geq 0, \theta \in \mathbb{R} & \end{array}$$

Let(x^v, θ^v) be an optimal solution where $(\theta_v)^T = (\theta_1^v, \ldots, \theta_K^v)^T$ is the vector of θ^k's. If $s_k = 0$ for some k, θ_k^v is ignored in the computation. If $c^T x^v + 1^T \theta^v = c^T a^v + \phi(a^v)$, stop; a^v is optimal.

Step 2. As before, if a feasibility cut is generated, set $a^{v+1} = a^v$ (null infeasible step), and go to Step 1.

*Step 3.*Same as the multicut version.

Step 4. If the inequality in step 3 does not hold for any k , then $a^{v+1} = x^v$ (exact serious step); go to Step 1.

Step 5. If $c^T x^v + \phi^v(x^v) \leq c^T a^v + \phi(a^v)$, then $a^{v+1} = x^v$ (approximate serious step); go to Step 1. Else, $a^{v+1} = a^v$ (null feasible step), go to *Step 0.*

1.4 Sample Average Approximation

Samples are generated using Monte Carlo Simulation, which implies expectation can be approximated as:

$$\mathbb{E}[Q(x,\xi)] \approx \frac{1}{N}\sum_{i=1}^{N} Q(x,\xi_i)$$

Based on this we can formulate the stochastic program approximately as

$$\begin{array}{lll} \min & c^T x + \frac{1}{N}\sum_{k=1}^{K} q_k y_k & \\ \text{subject to} & Ax = b & \\ & T_k x + W y_k = h_k, & k = 1,\ldots,K \\ & x \geq 0, y_k \geq 0, & k = 1,\ldots,K \end{array} \tag{3}$$

In the above equations, different instances of ξ are represented as those of $q_k, T_k \& h_k$. Algorithm:

Step 1. Initialization Choose N > 0, M > 0.

Step 2. Generate samples Generate M independent samples (batches) $\hat{\xi}^{1,v}, \ldots, \hat{\xi}^{N,v}$ according to the probability distribution of ξ, $v = 1,\ldots,M$, each of which has the sample-size N.

Step 3. Solve realizations of SAA For each of these samples solve the corresponding realization of (6), let $\nu_{N,v}$ be the optimal objective value, $v = 1,\ldots,M$.

*Step 4. Estimate f ** Use $\frac{1}{M}\sum_{v=1}^{M} \nu_{N,v}$ as an estimator of f * .

Step 5. Test the quality of solution This step involves statistical techniques for judging solution quality.

Samples generated via Monte Carlo generally have high variance, hence variance reduction techniques like importance sampling can be used to give better results.

1.5 Stochastic Decomposition

An alternative approach to using cuts produced with multiple samples in the L - shaped method is to use cuts constructed from small but increasing numbers of samples.

Algorithm:

Step 1 Set $v = 0, \xi^0 = \bar{\xi}$ and let x^1 solve

$$\min_{Ax=b, x\geq 0} \{c^T x + Q(x, \xi^0)\}$$

Step 2 Let $v = v + 1$ and let ξ^v be an independent sample generated from ξ. Find $\phi(x) = \frac{1}{v}\sum_{s=1}^{v} Q(x^v, \xi^s) = \frac{1}{v}\sum_{s=1}^{v}(\pi_s^v)^T(\xi^s - Tx^v)$. Let $E_v = \frac{1}{v}\sum_{s=1}^{v}(\pi_s^v)^T T$ and $e_v = \frac{1}{v}\sum_{s=1}^{v}(\pi_s^v)^T \xi^s$

Step 3. Update all previous cuts by $E_s \Leftarrow \frac{v-1}{v} E_s$ and $e_s \Leftarrow \frac{v-1}{v} e_s$ for $s = 1, \ldots, v-1$

Step 4. Solve the L-shaped master problem to obtain x^{v+1}. Go to step 2.

This is somewhat like an online version of the L-shaped algorithm. At each step we are increasing both the number of cuts and the number of samples. This works better than the standard SAA techniques in terms of faster convergence as is shown in [5].

1.6 Progressive Hedging

Progressive Hedging uses state aggregation techniques to construc the scenario tree and then solve the SP problem. Consider a scenario tree with ξ_s for $s \in S$ as the different scenarios, with each scenario being allocated a positive wieght p_s such that $\sum_{s\in S} p_s = 1$. This tree with corresponding probabilities is considered to be a representation of the generalized probability distribution. Using this rep, the expected value function can be calculated as $\mathbb{E}\{\phi(x, \xi)\} = \sum_{s\in S} p_s \phi(x, \xi_s)$.

Algorithm:

1. $k :== 0$
2. For all $s \in S, x_s^k := \arg\min_{x,y_s}(c^T x + f_s y_s) : (x, y_s) \in \Phi(x, \xi_s)$
3. $\bar{x}^k := \sum_{s\in S} p_s x_s^k$
4. For all $s \in S, w_s^k := \rho(x_s^k - \bar{x}^k)$
5. $k :== k + 1$
6. For all $s \in S, x_s^k := \arg\min_{x,y_s}(c^T x + f_s y_s + (w_s^{k-1})^T x + \frac{\rho}{2}||x - \bar{x}^{k-1}||) : (x, y_s) \in \Phi(x, \xi_s)$
7. $\bar{x}^k := \sum_{s\in S} p_s x_s^k$
8. For all $s \in S, w_s^k := w_s^{k-1} + \rho(x_s^k - \bar{x}^k)$
9. $\pi^k :=:= \sum_{s\in S} p_s ||x_s^k - \bar{x}^k||$
10. If $\pi^k > \epsilon$, then go to step 5, else terminate

Here in the minimization of the cost function in step 6, the cost function has changed to this because we are using the augmented lagrangian version of the problem. We penalize if the solution is far off from the average and we penalize the dot product of wieghts and the variables. This dot product being zero makes the policy implementable. By implementable we mean, it will then lie in one of the aggregated states. Hence, we penalize it being positive.

2 Solvers

2.1 PySP

PySP uses the Progressive Hedging algorithm to solve stochastic programs. To express a stochastic program in PySP, the user specifies both the deterministic base model and the scenario tree model with associated uncertain parameters in the Pyomo open-source algebraic modeling language. This separation of deterministic and stochastic problem components is similar to the mechanism proposed in SMPS. Historically, SMPS has served as the de facto interchange format for specifying stochastic programs—mirroring the early role of the MPS format in deterministic mathematical programming.

2.2 DAPPROX

DAPPROX implements the successive discrete approximation method, for complete recourse problems, with a deterministic objective in the second stage, and assuming the stochastic independence of the components of J. Probability distributions: finite discrete, uniform, exponential, and normal distributions.

3 Risk Measures

The most widely-adopted optimization criterion is represented by the risk-neutral expectation of a cumulative cost. However, in many applications one is interested in taking into account risk, i.e., increased awareness of events of small probability and high consequences. For e.g., in case of a portfolio optimization problem, if we consider the problem of maximising expected return rate, it's optimal solution suggests concentrating on the investment that maximizes return rate. This is not reasonable because it leaves out the risk of losing out all he invested money. Risk is especially important when considering problems where decisions are not taken again and again in which case the law of large numbers justifies us optmizing the expected value of a function. Generally speaking, risk can be taken into account in two ways; one being risk sensitive optimization wherein we quantify risk in the cost function itself and minimize it and the other is to include a constraint on the risk in our optimization problem.

Let $F : \mathbb{R}^n \times \Omega \to \mathbb{R}$. Consider for a feasible set $X \subset \mathbb{R}$. The risk neutral problem is given by

$$\min_{x \in X} \mathbb{E}\big[F(x, \omega)\big]$$

One of the classical approaches to taking risk into account in optimization is to define a utility function on the space of random variables.This idea is borrowed from expected utility theory of mathematical economics. Here, to compare two random outcomes, we compare the expected value of nondecreasing and convex scalar transformations $u : \mathbb{R} \to \mathbb{R}$ and prefer Z_1 over Z_2 if $\mathbb{E}[u(Z_1)] \leq \mathbb{E}[u(Z_2)]$. Using this method to incorporate risk in our problem, the risk averse optimization problem is given by

$$\min_{x \in X} \mathbb{E}\big[u(F(x, \omega))\big]$$

The transformation function $u(\cdot)$ here is also called the *disutility function*. This is used in minimization problems. In the maximization problems, we use a concave and non-decreasing scalar transform. In this case, it is called the *utility function*.

The main issue with this approach is specifying the utility function. There are no standard set of rules for this and using some utility functions we may get solutions which are difficult to interpret and explain. Hence, we move on to the Mean-Risk Models.

Mean Risk Models are inspired due to the Markowitz Mean-Variance Models. Characterize $F(x, \omega)$ by two scalar characterestics:

- $\mathbb{E}[Z]$: Describes expected outcome.
- $\mathbb{D}[Z]$: Measures the uncertainty of the outcome (risk dispersion measure)

Here the risk averse problem is given by:

$$\min_{x \in X} \mathbb{E}\big[Z_x\big] + c\mathbb{D}\big[Z_x\big]$$

The coefficient c here controls how much importance we want to give to the risk. Note that, here $\mathbb{D}$ can be variance, semideviations, weighted mean deviation from a quantile, etc. An issue with variance being a risk measure is it penalizes the random variable being less than the mean as much as it does so for it being greater than the mean. Hence, positive semideviation is a better risk measure.

Now, we move on to the current general way of incorporating risk using risk measures. Let $(\Omega, \mathscr{F})$ be a sample space equipped with the sigma algebra $\mathscr{F}$ on which considered uncertain outcomes (Random Functions Z) are defined.
Let the space of F-measurable functions be $\mathscr{Z}$.
A risk measure is a function $\rho(Z)$ which maps Z onto the extended real line. The Risk Averse Problem is now given by:

$$\min_{x \in X} \rho\big(F(x, \omega)\big)$$

To make use of such a definition, we need to apply some restrictions on the definitions. Such restrictions can be motivated by what we observe in real life or can be so to make the problem easier to optimize. Hence, we define *Coherent measures of Risk*. Let (Ω, F, P) be a probability space, $X : \Omega \to \mathbb{R}$ denote a certain outcome, and Z be the space of possible outcomes, $\mathscr{Z} = L_p(\Omega, F, P)$. Let the random outcome represent losses. Then a coherent risk measure is defined to be a risk measure having the following properties:

- **(P1)** Monotonicity: If Z, Z'$\in \mathscr{Z}$ and $Z \geq Z'$, then $\rho(Z) \geq \rho(Z')$

- **(P2)** Translational Equivariance: If Z$\in \mathscr{Z}$ and $a \in \mathbb{R}$, then $\rho(Z + a) = \rho(Z) + a$
- **(P3)** Positive Homogeneity: If Z$\in \mathscr{Z}$ and $t > 0$, then $\rho(tZ) = t\rho(Z)$
- **(P4)** Convexity: If Z, Z'$\in \mathscr{Z}$ and $t \in [0, 1]$, then

$$\rho(tZ + (1 - t)Z') \leq t\rho(Z) + (1 - t)\rho(Z')$$

The above formulation was proposed by Artzner, et al in [1] in the context of capital risk. Hence, the motivation for each of the above properties have roots in mathematical finance.
Monotonicity: The intuition behind this is that investing more capital under any possible market scenario increases the investor's perception of risk. Also, it rules out some badly behaved risk measures such as $\rho(X) = \mathbb{E}[X] + c\sigma(X)$, where σ denotes the standard deviation operator.
Translational Equivariance: Intuition is that investing an extra amount increases the perception of risk by the same amount.
Convexity and Positive Homogeneity: Together these imply subadditivity, which implies "A merger does not create extra risk". If the risk were to fail to satisfy this property, then for someone wishing to take a risk of $X_1 + X_2$, it would be better to open two accounts, one with risk X_1 and other with risk X_2, incurring the risk of $\rho(X_1) + \rho(X_2)$ which would be less than or equal to $\rho(X_1 + X_2)$.

Now we move on to specific risk measures. We will focus on the Value at Risk(VaR_α) and the Conditional Value at Risk ($CVaR_\alpha$). VaR_α is also called the chance constraint. It is defined as:

$$\beta_\alpha(X) = arg\inf_\beta\{\beta : \mathbb{P}(X > \beta) \leq \alpha\}$$

The above measure is not convex and hence not a coherent risk measure. Defining $CVaR_\alpha$ can be thought of in many different ways. It's definition is:

$$CVaR_\alpha(X) = \mathbb{E}[X|X > \beta_\alpha]$$

It is shown in [2] that the conditional value at risk is the solution of the minimization problem:

$$CVaR_\alpha[Z] = \inf_{u\in\mathbb{R}} \left\{u + (1 - \alpha)^{-1}\mathbb{E}[Z - u]_+\right\} \tag{4}$$

Also, [6] gives a constructive treatment of how to reach to the above minimization problem by considering $CVaR_\alpha$ as a convex approximation of VaR_α.

4 Problem Formulation and Solution

We define the problem as given in the proposal as

$$\begin{aligned}
\min_{z_i, r_i, \theta_i} \quad & \rho_\alpha\Big(\sum_i c_i^1(z_i) + c_i^2(r_i)\Big) \\
\text{subject to} \quad & \sum_{m=1}^{N} Y_{nm}(\theta_n - \theta_m) = \sum_{i\in G_n}(z_i + r_i) - D_n, && \forall n, m \\
& Y_{nm}(\theta_n - \theta_m) \leq C_{nm}, && \forall n, m \\
& r_i \in [-\underline{R_i}, \overline{R_i}], && \forall i \\
& z_i + r_i \in [\underline{G_i}, \overline{G_i}], && \forall i
\end{aligned}$$

where D_n is the net demand at node n, G_n is the set of generators at node n, z_i is the generation dispatch of generator i, with cost $c_i^1(z_i)$, r_i is ramping dispatch in stage II (just before demand fulfillment) with ramping cost $c_i^2(r_i)$, $\underline{G_i}$ and $\overline{G_i}$ denoting the minimum and maximum generation capacities of generator i, and $-\underline{R_i}$ and $\overline{R_i}$ are the corresponding ramping constraints, ρ_α is a risk measure.

The reformulated problem has 2 more variables per bus introduced. One is for the changes in the phase angle during changes due to ramping represented by ϕ_i, the other is for the power balance generators which are present at each node where there is already a generator. This is specifically added to make the problem feasible under all circumstances. We will be putting a very high cost for using these generators. The variable representing them is b_i.

Reformulated problem:

$$\begin{aligned}
\min_{z_i,r_i,\theta_i,\phi_i,b_i} \quad & \rho_\alpha\Big(\sum_i c_i^1(z_i) + c_i^2(r_i) + c_i^3(b_i)\Big) \\
\text{subject to} \quad & \sum_{m=1}^{N} Y_{nm}(\theta_n + \phi_n - \theta_m - \phi_m) = \sum_{i\in G_n}(z_i + r_i + b_i) - D_n, \quad \forall n, m \\
& Y_{nm}(\theta_n + \phi_n - \theta_m - \phi_m) \le C_{nm}, \quad \forall n, m \\
& r_i \in [-\underline{R_i}, \overline{R_i}], \quad \forall i \\
& z_i + r_i \in [\underline{G_i}, \overline{G_i}], \quad \forall i
\end{aligned}$$

We divide this problem into two stages. The variables in the first stage are z_i, θ_i, whereas those in the second stage are r_i, ϕ_i and b_i, D_n is the random variable, on observation of which the second stage decision is taken. We want to consider the CVaR risk measure here. Instead of choosing ρ_α directly as $CVaR_\alpha$ we can define it as:

$$\rho_{\alpha,\lambda}[Z] \coloneqq (1-\lambda)\mathbb{E}[Z] + \lambda CVaR_\alpha[Z] \tag{5}$$

where $\lambda \in [0, 1]$. This gives us the freedom of how much importance do we want to give to the risk by choosing λ.

Now, following the development in [3], our problem can be written using the above risk measure as:

$$\begin{aligned}
\min_{z_i,r_i,\theta_i,\phi_i,b_i,u} \quad & \sum_i c_i^1(z_i) + \lambda u + \mathbb{E}\big\{(1-\lambda)\big(\sum_i c_i^2(r_i) + c_i^3(b_i)\big) \\
& + \quad \lambda(1-\alpha)^{-1}\big[\big(\sum_i c_i^2(r_i) + c_i^3(b_i)\big) - u\big]_+\big\} \\
\text{subject to} \quad & \sum_{m=1}^{N} Y_{nm}(\theta_n + \phi_n - \theta_m - \phi_m) = \sum_{i\in G_n}(z_i + r_i + b_i) - D_n, \quad \forall n, m \\
& Y_{nm}(\theta_n + \phi_n - \theta_m - \phi_m) \le C_{nm}, \quad \forall n, m \\
& r_i \in [-\underline{R_i}, \overline{R_i}], \quad \forall i \\
& z_i + r_i \in [\underline{G_i}, \overline{G_i}], \quad \forall i
\end{aligned}$$

We have introduced an additional variable u and changed the CVaR constraint into an expectation. Now it is possible to split the problem into two stages easily as follows:

Recourse problem: $P_1(z, \theta, u, D)$

$$
\begin{aligned}
\min_{r_i,\phi_i,b_i} \quad & (1-\lambda)(\sum_i c_i^2(r_i) + c_i^3(b_i)) + \lambda(1-\alpha)^{-1}[(\sum_i c_i^2(r_i) + c_i^3(b_i)) - u]_+ \\
\text{subject to} \quad & \sum_{m=1}^{N} Y_{nm}(\theta_n + \phi_n - \theta_m - \phi_m) = \sum_{i \in G_n} (z_i + r_i + b_i) - D_n, && \forall n, m \\
& Y_{nm}(\theta_n + \phi_n - \theta_m - \phi_m) \leq C_{nm}, && \forall n, m \\
& r_i \in [-\underline{R_i}, \overline{R_i}], && \forall i \\
& z_i + r_i \in [\underline{G_i}, \overline{G_i}], && \forall i
\end{aligned}
$$

This can be written as a linear program as follows:

$$
\begin{aligned}
\min_{r_i,\phi_i,b_i,v} \quad & (1-\lambda)(\sum_i c_i^2(r_i) + c_i^3(b_i)) + \lambda(1-\alpha)^{-1} v \\
\text{subject to} \quad & \sum_{m=1}^{N} Y_{nm}(\theta_n + \phi_n - \theta_m - \phi_m) = \sum_{i \in G_n} (z_i + r_i + b_i) - D_n, && \forall n, m \\
& Y_{nm}(\theta_n + \phi_n - \theta_m - \phi_m) \leq C_{nm}, && \forall n, m \\
& r_i \in [-\underline{R_i}, \overline{R_i}], && \forall i \\
& z_i + r_i \in [\underline{G_i}, \overline{G_i}], && \forall i \\
& (\sum_i c_i^2(r_i) + c_i^3(b_i)) - u \leq v
\end{aligned}
$$

Note that in the above problem, z_i, θ_i, u and D_n are parameters. Solution of this can be represented as:

$$\phi(z, u, \theta, D) = inf\{P_1(z, \theta, D)\}$$
$$\Phi(z, u, \theta, D) = \arg\min\{P_1(z, \theta, D)\}$$

Next, we define

$$\overset{\wedge}{\phi}(z, u, \theta) = E_D[\phi(z, \theta, D)]$$

The projected problem is:

$$\min_{z_i,\theta_i,u} \quad \sum_i c_i^1(z_i) + \lambda u + \overset{\wedge}{\phi}(z, u, \theta)$$

To the above projected problem, we can apply the constraint that z_i and θ_i are selected so that the power flow equation is satisfied for the expected demand i.e.

$$\sum_{m=1}^{N} Y_{nm}(\theta_n - \theta_m) = \sum_{i \in G_n} (z_i) - \overline{D}_n, \forall n, m$$

References

[1] P. Artzner, F. Delbaen, J.-M. Eber, and D. Heath. Coherent measures of risk. *Mathematical Finance*, 9:203–228, 1999.

[2] R. T. Rockafellar and S. P. Uryasev, "Optimization of conditional value-at-risk", *The Journal of Risk*2, 21–41, 2000.

[3] A. Shapiro. " Analysis of stochastic dual dynamic programming method" *European Journal of Operational Research*, 209:63–72, 2011.

[4] J. R. Birge and F. Louveaux " Introduction to Stochastic Programming", *Springer Series in Operations Research and Financial Engineering*, 2011

[5] J. Higle, "Stochastic Programming: Optimization When Uncertainty Matters", *Tutorials in Operations Research, INFORMS* 2005.

[6] Shapiro, A., & Philpott, A. (2007). A tutorial on stochastic programming, Manuscript. http://www2.isye.gatech.edu/ ashapiro/publications.html

[7] A. Shapiro, D. Dentcheva, A. Ruszczynski, Lectures on Stochastic Programming: Modeling and Theory, SIAM, Philadelphia, 2009