Models and Simulation for Bulk Gas Production and Distribution

Wasu Glankwamdee
Jeff Linderoth
Jierui Shen

ISE Department
Lehigh University

COR@L
COMPUTATIONAL OPTIMIZATION RESEARCH AT LEHIGH

Jim Hutton
Peter Connard
Air Products

AIR PRODUCTS

EWO Spring Meeting
Pittsburgh, Pennsylvania
March 14, 2007

In Our Last Episode(s)...

- Our project is studying the impact of uncertainty in AP's production-distribution planning process.
- Uncertainty comes from
  - Customer demand
  - Production quantity (i.e. plant may go down)
  - Competitor removal amounts (regional swap contracts)
- We've built on optimization model that allows for different time-aggregations of the supplies and demands

---

- Building a model is really only a first step.
- Model can be tested and validated with simulation

Entities in the Production/Distribution Simulation

- Sites
  - Location, Production Capacity and Costs, ...
- Customers
  - Location, Demand
- InstanceFamily
  - Sites, Customers
  - Planning Horizon (e.g. 28 days)
- Instance
  - Time-aggregated optimization instance solve via XPRESS-MP

Using the Optimization Model

- The Optimization model contains really two classes of variables that we care about...

Optimization Variables Used By Simulation

1. Production Decisions x_{pst}: Production of product p at site s during period t.
   - For Simulation: we only care about this for period $t = 1$
2. Sourcing Decisions y_{psct}: Amount of product p delivered from site s to customer c during period t
   - For Simulation: Don't care about quantites – only sourcing decisions. (Deliveries typically made in full truckload amounts.)
   - We only care about the $t = 1$ sourcing decisions

Evaluating a Production/Distribution Policy

0. $t = 0$. Generate an instance with initial input and forecasts.
1. Generate a solution (policy) (x, y):
2. Re-sample data (demand).
3. Simulate (x, y) for some simulation increment Δ, resulting in a "real" cost c_t.
4. Create new forecasts based on current initial conditions, generate a new instance with different input state. Let $t = t + 1$. Goto 1.

How to Simulate

- The simulation (step 3 in previous slide) is admittedly quite rudimentary.

Our "Simulation"

1. Produce at rate suggested by solution (Production decision x) for "simulation increment" amount of time.
2. Make deliveries. Deliveries are made to customers c whose expected inventory level will fall below a threshold.
3. Deliveries are made only from sites s for which $y_{psct} > 0$: (Sourcing Decision from optimization model)
4. Deliveries are made in full truckloads. (Routing ignored).

Last Time: What We Said was "Next"

An actual slide from Last November's presentation...

Transfer Technology to Air Products

- Run on Real Data
  - Need to write code to instantiate objects from data files

Keep Considering Uncertainty!

- Minimax Model
- Robust Optimization Model
- Stochastic model
- Disruption Planning: Consider more "extreme" scenarios. Help build "contingency lists" for sourcing decisions.

To Do Item #1: Running on Real Data

- We identified a reasonable case study location.
- We had a celebratory lunch at a fish-and-chips place in Bethlehem.
- We decided on the input file format for data transfer
- Simulation/optimization code was instrumented to instantiate from these data files
- One (fairly interesting(?)) wrinkle. How to easily convey/pass distribution information?

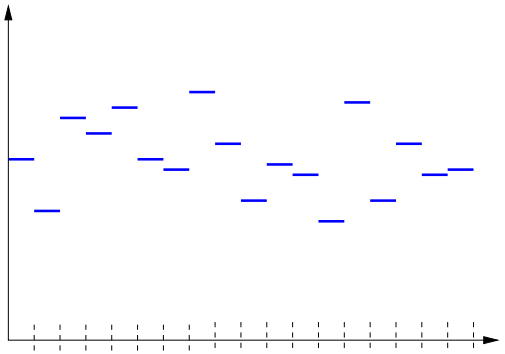
Customer Flavors

Customer's Demands Might Be...

1. Normally Distributed
2. On-Off
3. Specific Needs
4. Call-In

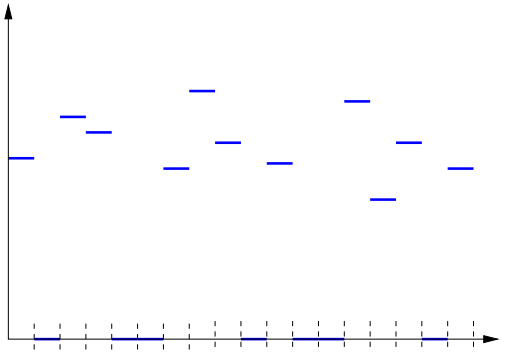
Normally Distributed

- Daily demand is $\mathcal{N}(\mu, \sigma^2)$.
- Daily demand is independent.



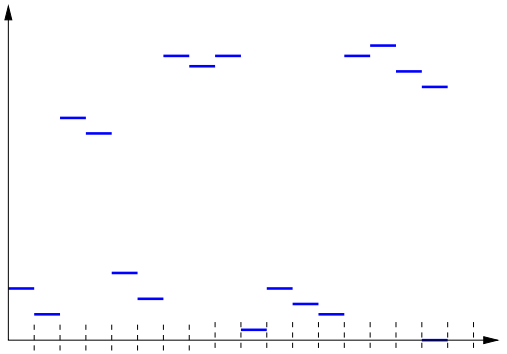
Zero Or Normally Distributed (On-Off)

- With probability $p_t > 0$, daily demand is 0.
- Else, daily demand is $\mathcal{N}(\mu, \sigma^2)$.
- Daily demand is independent.
- Real case: p_t may be known ($p_t = 1$ if t is weekend).



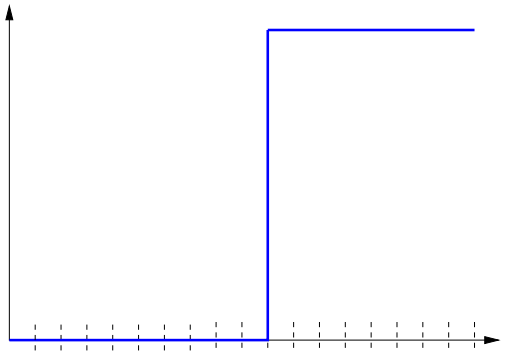
Specific Needs

- Demand is typically $\mathcal{N}(\mu, \sigma^2)$.
- But at certain points in time, the customer try a new process, requiring significantly more consumption.
- Daily demand is not independent.



Call-In

- Demand is typically 0.
- But at certain points in time, the customer requires significant (nearly constant) daily consumption.
- Examples: Cranberry season. Flash-freezing fish harvest.
- Daily demand is not independent.



Ain't Nothin' Easy When You're Doin' It Fer Real

- These flavors were a rather "loose" characterization of the customers
- AP may not exactly have customers in these categories
- Further, how should AP pass to the simulation information about how the (customer) distribution should look as a function of time?

The Idea

- Random "stuff" (stochastic process) $\{\xi_t\}$ happens
- Overlay each customer's daily, independent demands with an "impulse" function of the random vectors: $\mathcal{I}_c(\xi_t)$

$$d_{ct}(\xi_t) = \max(0, \mathcal{N}(\mu_c, \sigma_c^2) + \mathcal{I}_c(\xi_t)).$$

Using the Impulse

- Normally Distributed: $\mathcal{I}_c(\xi_t) = 0$
- On-Off: $\mathcal{I}_c(\xi_t) \in \{0, -\infty\}$
- Special Needs: $\mathcal{I}_c(\xi_t) \in \{0, m_t\}$
- Call In: $\mu_c = \sigma_c^2 = 0$, $\mathcal{I}_c(\xi_t) \in \{0, m_t\}$

- AP will specify $\mathcal{I}_c(\cdot)$ as scenario trajectories of the impulse value per day

Customer Gassy

- Scenario 1, probability 1/4, $\mathcal{I}_{\text{Gassy}} = (0, 0, -\infty, -\infty, 0, \ldots)$
- Scenario 2, probability 1/4, $\mathcal{I}_{\text{Gassy}} = (0, -\infty, -\infty, 0, 0, \ldots)$
- Scenario 3, probability 1/2, $\mathcal{I}_{\text{Gassy}} = (-\infty, -\infty, -\infty, -\infty, 0, \ldots)$

To Do Item #2: Keep Considering Uncertainty

- Recall that our `Instance` class is a representation of an optimization instance: it is something that can be solved.
- We would like to consider different "ways" of solving (different policies)
  - Deterministic (at various time-grains)
  - Stochastic (different numbers of stages)
  - Robust
  - Minimax: minimize the maximum cost in a number of outcomes
- We made the `Instance` class abstract.

Instance Class

```
virtual void writeMosel(std::string &fileName) = 0;
virtual int setSolution(XPRMmodel &ewoMod) = 0;

std::vector<ProductionAmount> extProd_;
std::vector<ProductionAmount> regProd_;
std::vector<Delivery> deliveries_;
std::vector<CustomerShortage> custShortage_;
```

- `writeMosel`: Write out data files that Mosel models expect
- `SetSolution`: Must read solution variables out of XPRESS, and place policy's suggested production and delivery amounts into the base class containers.
  - Later this method will be extended to read solutions from a file, as XPRESS's stochastic solver can't handle realistic-sized instances in this case.

Current Status

- Mosel code for stochastic and minimax models written
- Hooks to optimization/simulation code for these models "in progress."
  - Wasu to the Rescue!
- XPRESS SP solver will not work for this instance. Use special-purpose solver for large-scale problems
  - Jerry to the Rescue!
- Data from Air Products is "in the mail."

Conclusions

- Lehigh and Air Products Don't Work Very Hard over the holidays. :-)
- We're well on our way to having a system that is flexible enough to handle our prototype case study
- Deployment:
  - ϵ version of simulation/optimization code turned over to AP
  - Jerry will go to work at Air Products. He will also continue to work on simulation/optimization tools when he is there.
- Publication: We're aiming for Kevin and Ignacio's Special Issue of *Computers and Chemical Engineering*