# ASK THE EXPERT

## Composite AI Using Forecasting and Optimization

**Jay Laramore**
Senior Analytical Training Consultant

**Rob Pratt**
Senior R&D Manager

# Jay Laramore

Senior Analytical Training Consultant

Jay Laramore has worked at SAS since 2015. He currently develops and teaches education courses in econometrics, time series forecasting, network analysis and optimization. Previously, Jay worked in the analytical consulting division at SAS, where he built and deployed a variety of analytical solutions for customers. Prior to joining SAS, he worked as an analytical consultant in the marketing and retail fuel industries. He earned a master's degree in economics from the University of North Carolina at Greensboro and a bachelor's degree in economics from Roanoke College.

# Rob Pratt

Senior R&D Manager

Rob Pratt has worked at SAS since 2000 and is a Senior Manager in the Scientific Computing department in the Analytics R&D division. He manages a team of developers responsible for the optimization modeling language and solvers for linear, mixed integer linear, quadratic, and conic optimization. He earned a B.S. in Mathematics (with a second major in English) from the University of Dayton and both an M.S. in Mathematics and a Ph.D. in Operations Research from The University of North Carolina at Chapel Hill.

# Composite AI Using Forecasting and Optimization

## Introduction

- Composite AI problems combining forecasting and optimization can be used to satisfy several objectives:
  I. Anticipate and satisfy customer demand
  II. Maintain acceptable inventory balances
  III. Minimize total operational cost across all suppliers
  IV. Accommodate problem-specific business rules that influence decision making

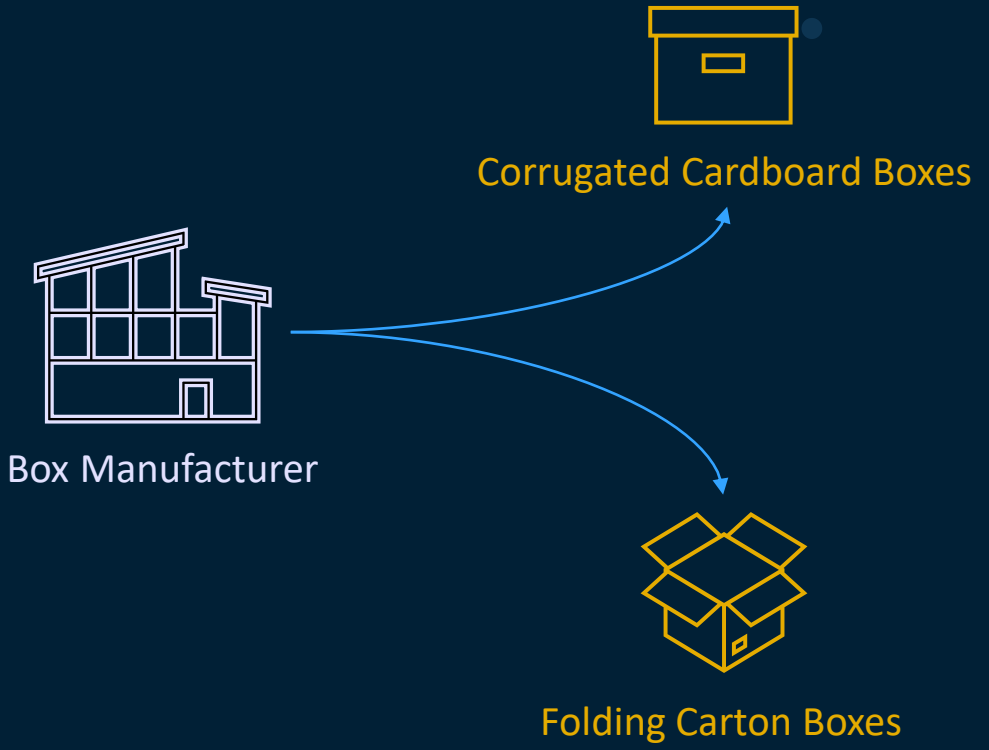
- Examples:
  - Call center staffing
  - ATM replenishment
  - Retail fuel restocking
  - Restaurant seating optimization
  - Retail product restocking and replenishment
  - Marketing revenue optimization

# Composite AI Using Forecasting and Optimization

## Business Case Study: A Box Manufacturer

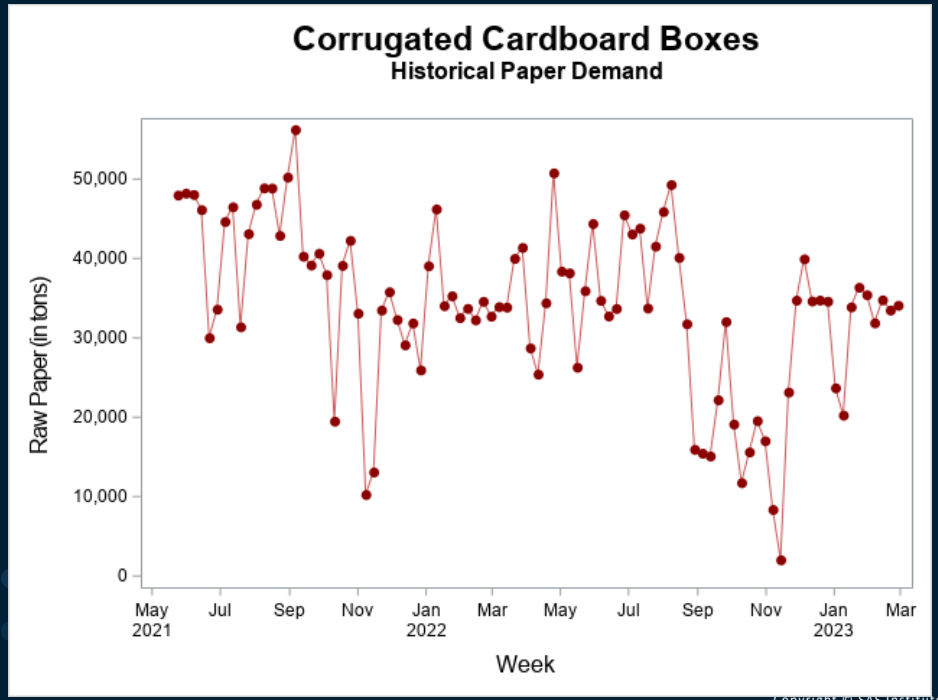
- Every week, a **Box Manufacturer** produces and distributes two (2) types of boxes:
  I. Corrugated Cardboard Boxes
  II. Folding Carton Boxes

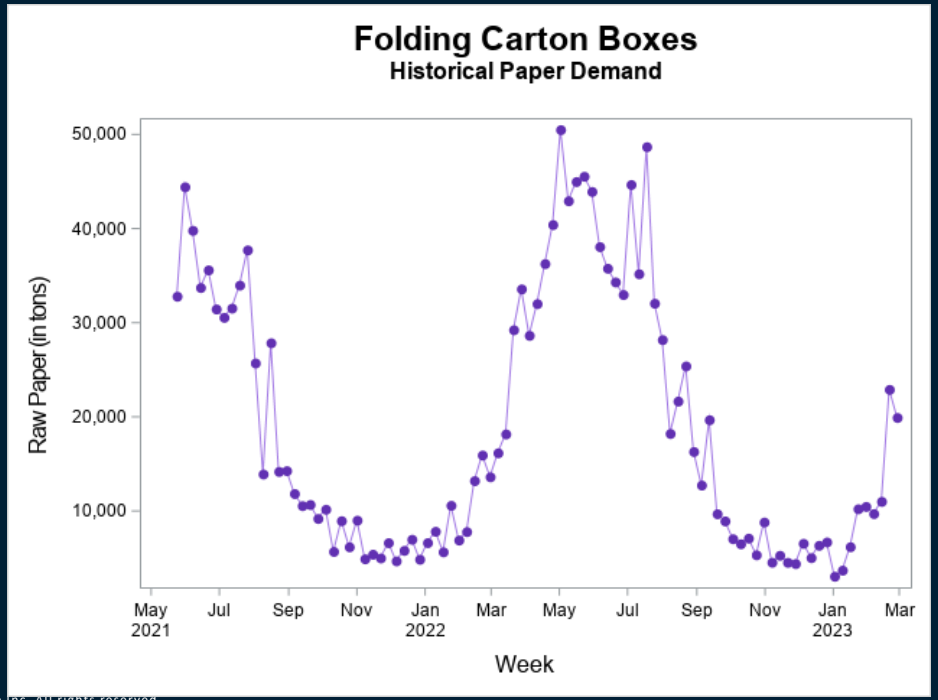
- The **Box Manufacturer** is responsible for satisfying customer demand each week for all box types



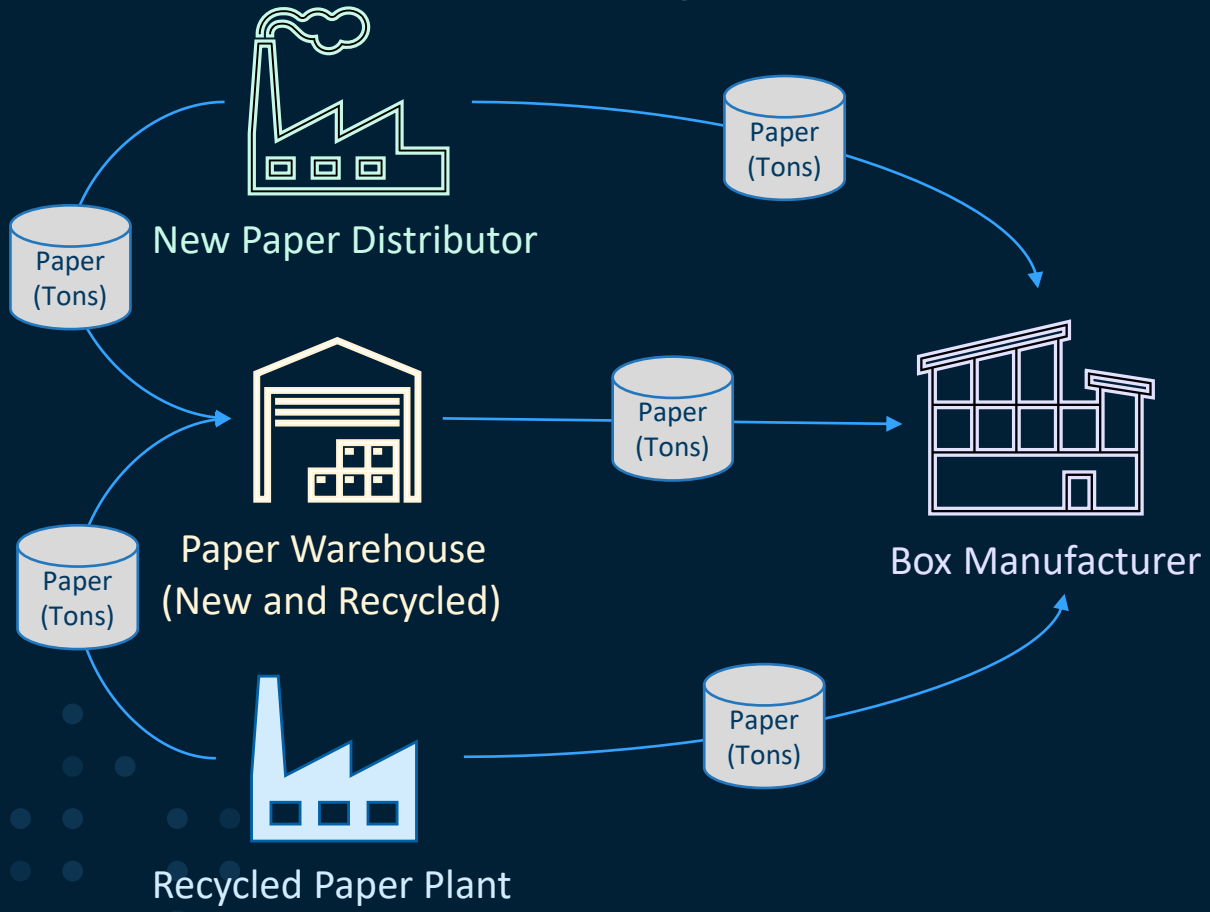
# Historical Paper Demand By Box Type

- To anticipate future production levels, historical demand for each box type is used to forecast customer demand and raw material requirements for the next four weeks



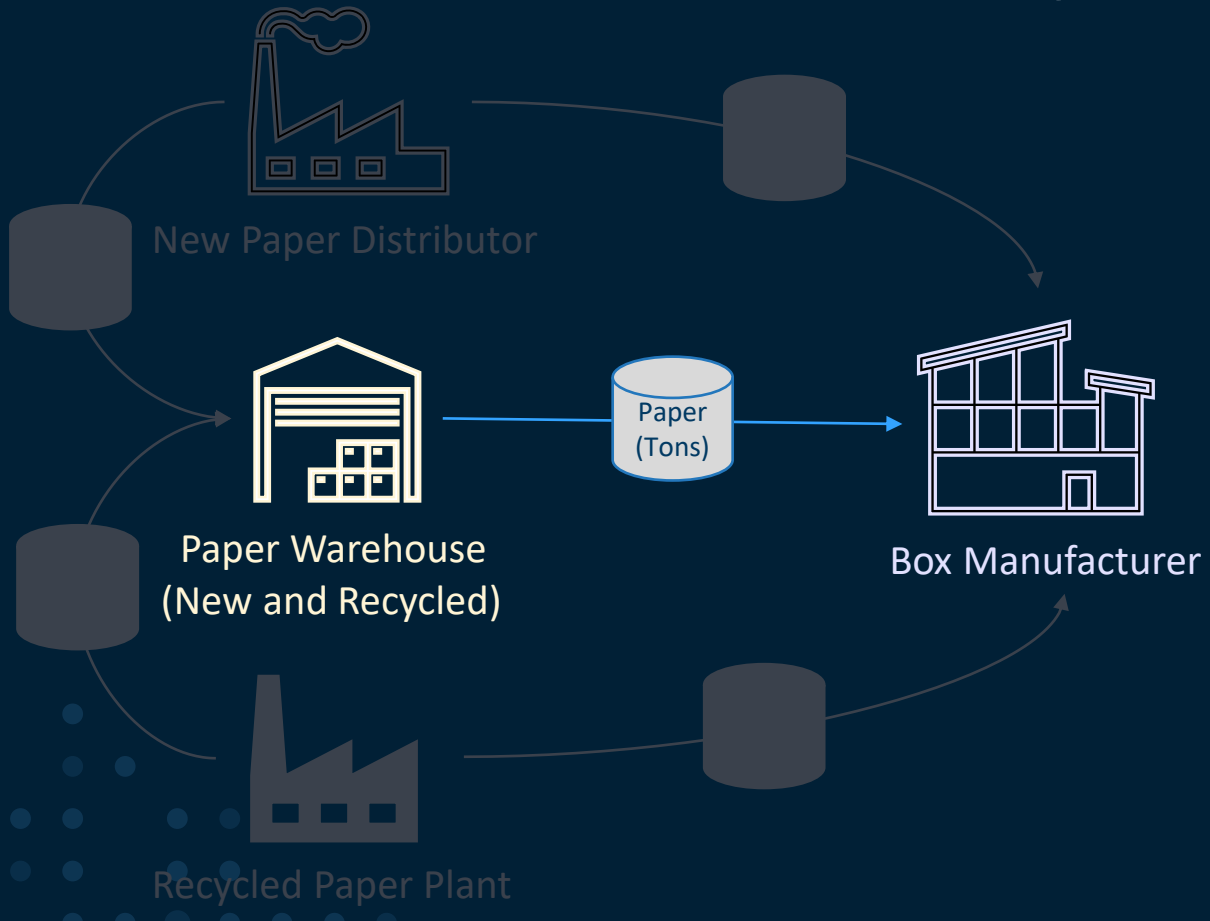
# Weekly Business Model Diagram




- To manufacture boxes, each week the **Box Manufacturer** sources its paper from two suppliers:
  I. An external vendor named **New Paper Distributor**
  II. Its company-owned **Recycled Paper Plant**

- The **Box Manufacturer** can also use paper currently stored in its **Paper Warehouse**

# Current Inventory Balance

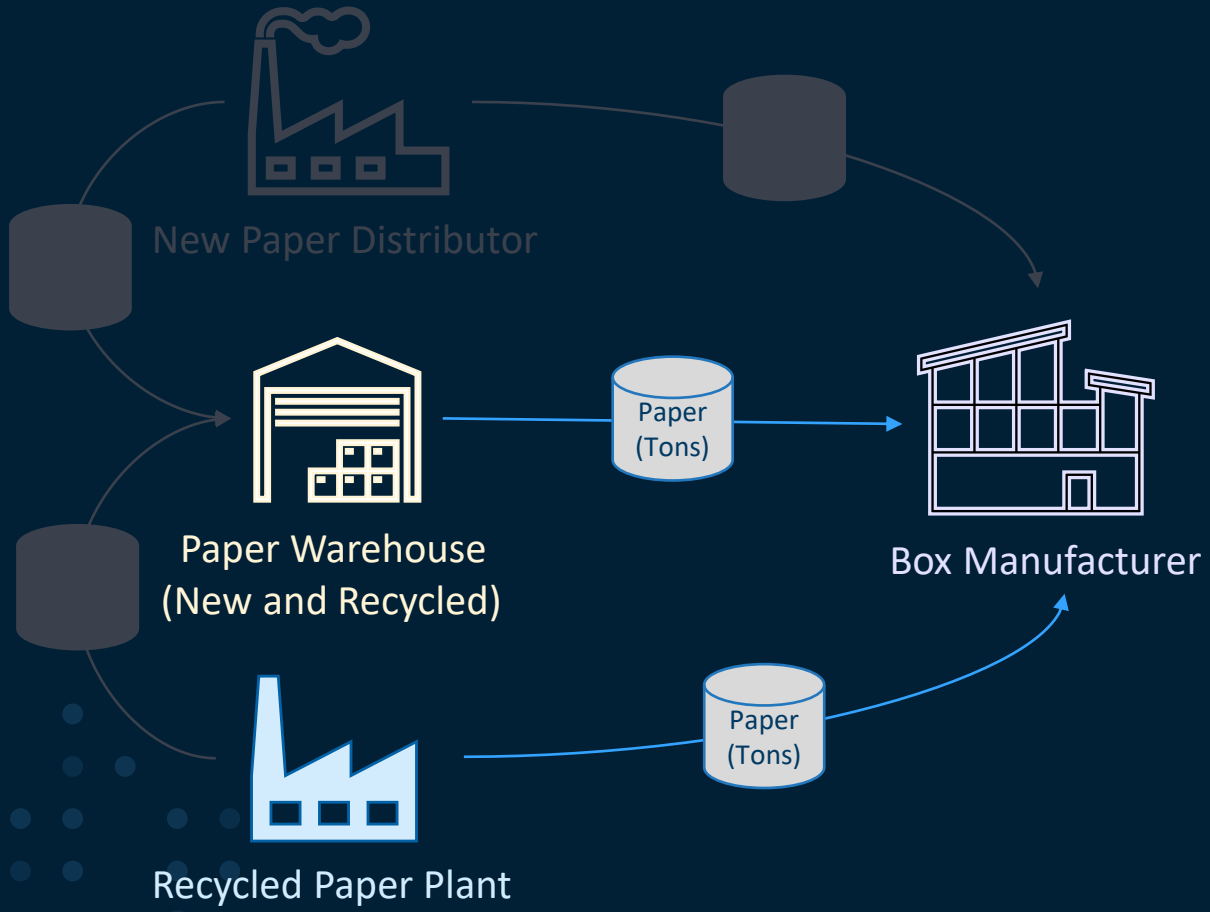



- Current inventory levels in the **Paper Warehouse**:
  - New Paper: 45,000 tons
  - Recycled Paper: 62,500 tons
- For simplicity, assume no cost for storage, transfer, etc.
- Because the inventory was purchased in prior weeks, it can be used cost-free in current or future weeks to produce boxes

# Weekly Recycled Paper Production Schedule

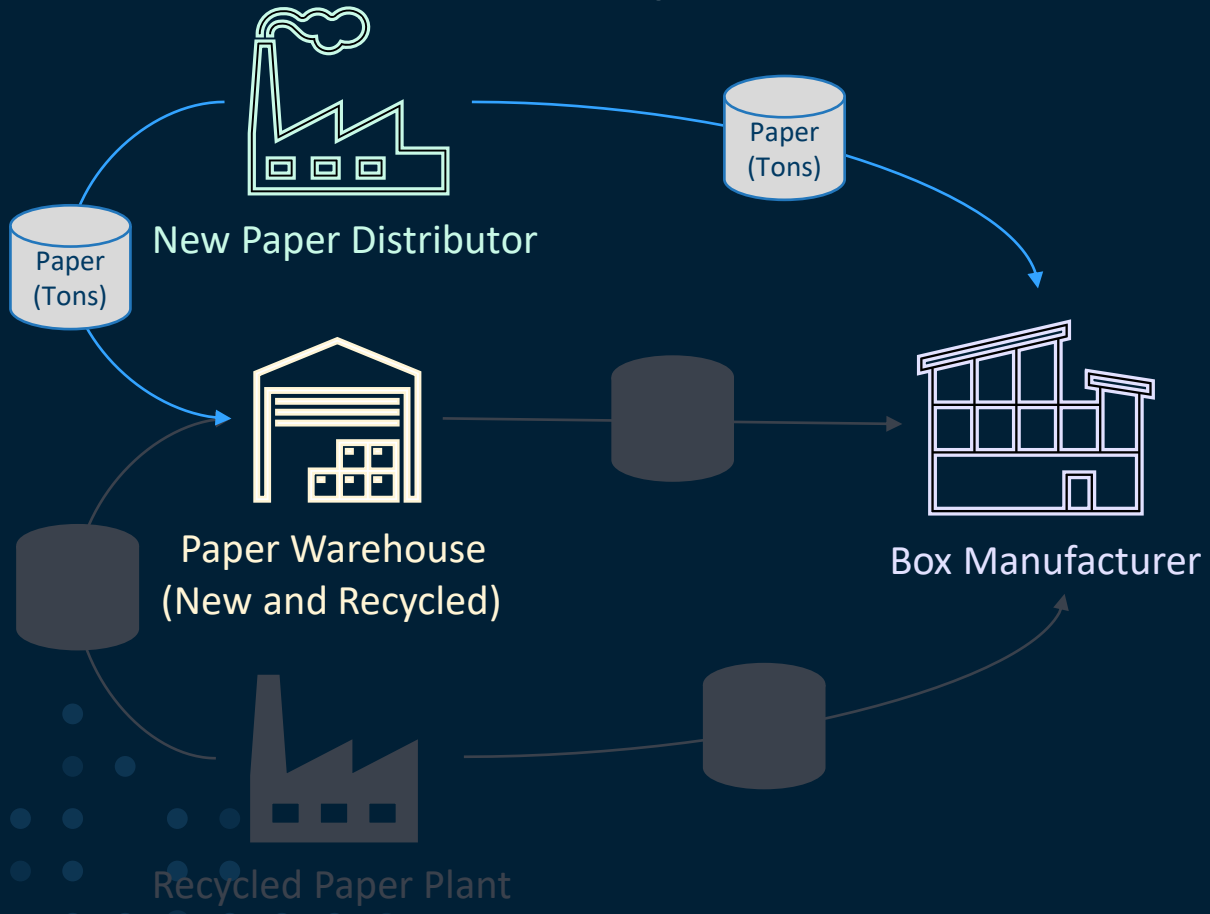

- For the next four weeks, the **Recycled Paper Plant** has committed to the following production schedule:
  - **t+1**: 12,000 tons, $0.10/ton
  - **t+2**: 18,000 tons, $0.10/ton
  - **t+3**: 20,000 tons, $0.18/ton
  - **t+4**: 22,000 tons, $0.18/ton
- Recycled paper produced in the current week can be used for box production, or moved to the **Paper Warehouse** for storage

# Environmental Business Requirement



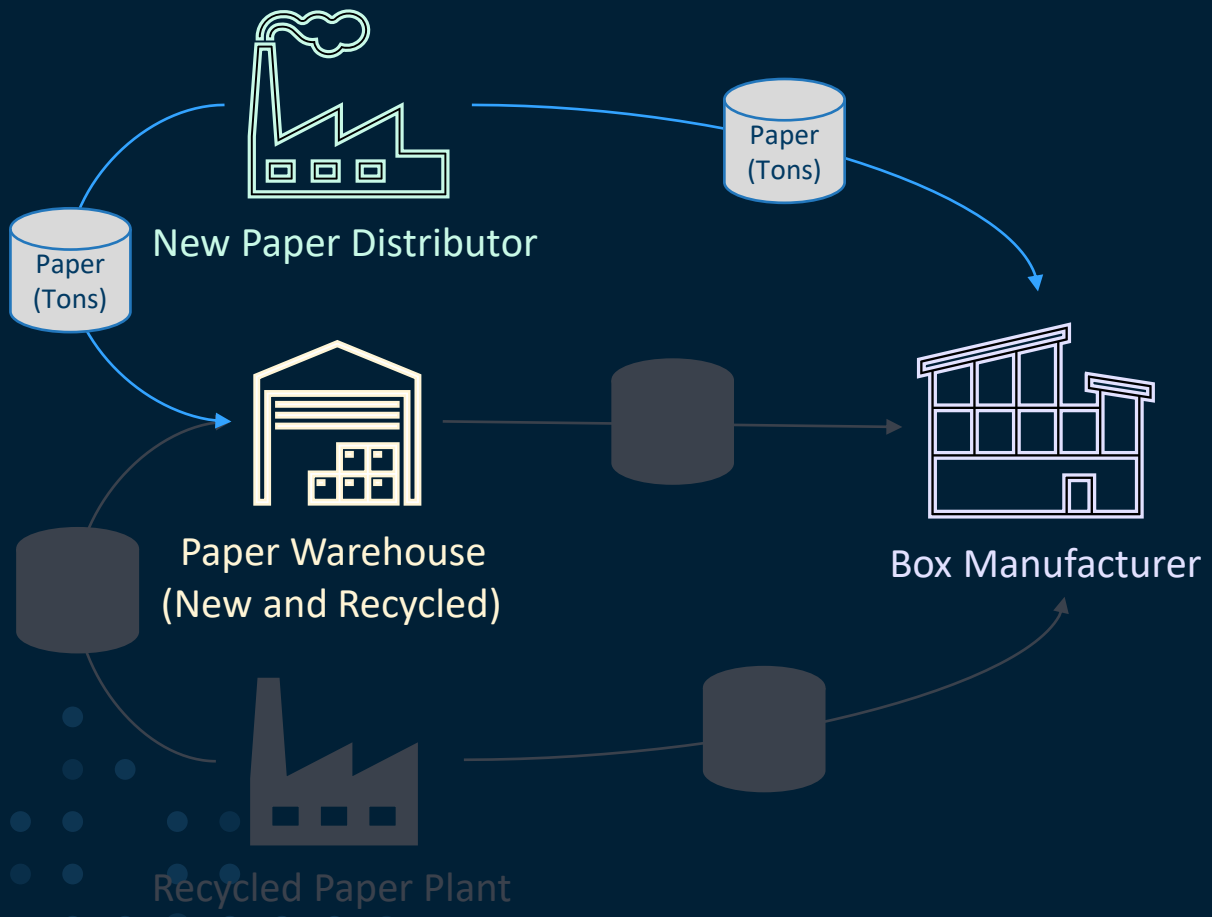

- Each week, the **Box Manufacturer** must use at least 25% recycled paper to manufacture its boxes
- The recycled paper can either come directly from the **Recycled Paper Plant** or the **Paper Warehouse**

# New Paper Distributor Contract Tiers




- The **Box Manufacturer** can purchase new paper each week at a contracted price-per-ton rate.
- Purchase contracts between the **Box Manufacturer** and the **New Paper Distributor** renew every four weeks, and it's time to renew the contract for the next four weeks

# New Paper Distributor Contract Tiers




| Tier | Minimum Tons Purchased Per Week | Price Per Ton |
|---|---|---|
| Tier 1 | ≥ 10,000 | $0.15 |
| Tier 2 | ≥ 25,000 | $0.12 |
| Tier 3 | ≥ 35,000 | $0.09 |
| Tier 4 | ≥ 50,000 | $0.07 |

- Based on the box demand forecasts, current new and recycled inventory levels, the **Recycled Paper Plant** production schedule, and the 25% recycled paper requirement each week, which tier minimizes total operational cost over the next four weeks?

# SAS® Visual Forecasting

## Key Capabilities

### AUTOMATION

Automatically generate, manage, and deploy large numbers of trustworthy forecasts using a future-proof solution you can trust and configure.

### CUTTING-EDGE TECHNIQUES

Employ time-series, machine-learning, hybrid (ML + time series), and deep learning techniques to improve your forecasting accuracy.

### EXTERNAL DRIVERS

Incorporate events, holidays, and external drivers and let the system automatically choose which are important.

### EMPOWER OPEN-SOURCE

Scale open-source algorithms to run in parallel in the cloud, and in a consistent framework.

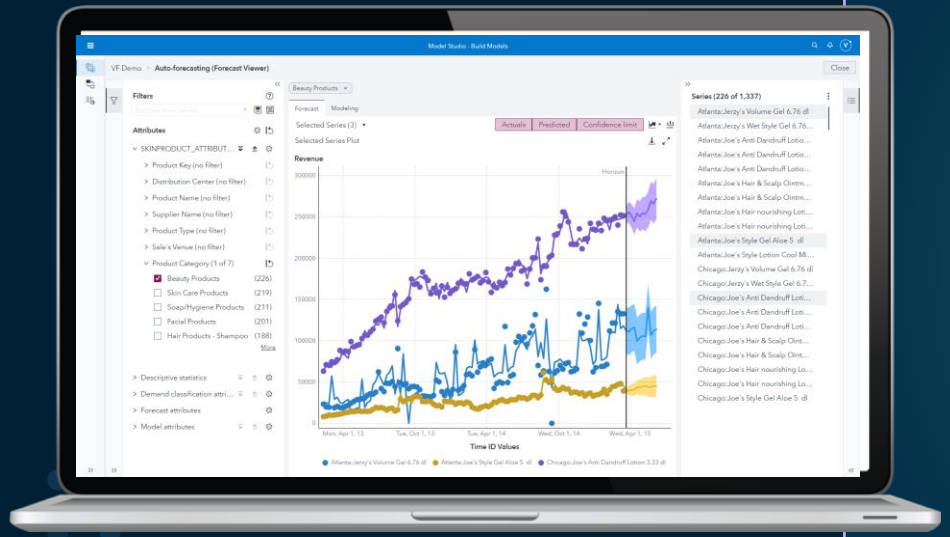
### BUSINESS KNOWLEDGE

Using your business knowledge, apply overrides in a flexible manner.

### CUSTOMER SUPPORT

Get the support you need to succeed by taking SAS forecasting courses and joining a vibrant forecasting community.

# SAS Visual Forecasting:
# Automatically Generate Trustworthy Forecasts at Scale




- Supports **full analytics** life cycle
- Performs **automatic large-scale** forecasting
- Choice of **no/low code forecasting** via visual interface as well as the **programming option** via SAS or open APIs
- Interactive **forecasting viewers**
- Interactive **modeling capabilities**
- **Automatic outlier detection** and treatment
- Create your own **reusable components** to share

# SAS® Visual Forecasting Screenshots

| Obs | box_type | date | paper_demand |
|---|---|---|---|
| 1 | Folding Carton | Mon, 31 May 2021 | 32,842 |
| 2 | Folding Carton | Mon, 7 Jun 2021 | 44,470 |
| 3 | Folding Carton | Mon, 14 Jun 2021 | 39,841 |
| 4 | Folding Carton | Mon, 21 Jun 2021 | 33,769 |
| 5 | Folding Carton | Mon, 28 Jun 2021 | 35,640 |
| 6 | Folding Carton | Mon, 5 Jul 2021 | 31,490 |
| 7 | Folding Carton | Mon, 12 Jul 2021 | 30,596 |
| 8 | Folding Carton | Mon, 19 Jul 2021 | 31,583 |
| 9 | Folding Carton | Mon, 26 Jul 2021 | 34,026 |
| 10 | Folding Carton | Mon, 2 Aug 2021 | 37,763 |
| 11 | Folding Carton | Mon, 9 Aug 2021 | 25,751 |
| 12 | Folding Carton | Mon, 16 Aug 2021 | 13,956 |
| 13 | Folding Carton | Mon, 23 Aug 2021 | 27,897 |
| ... | ... | ... | ... |

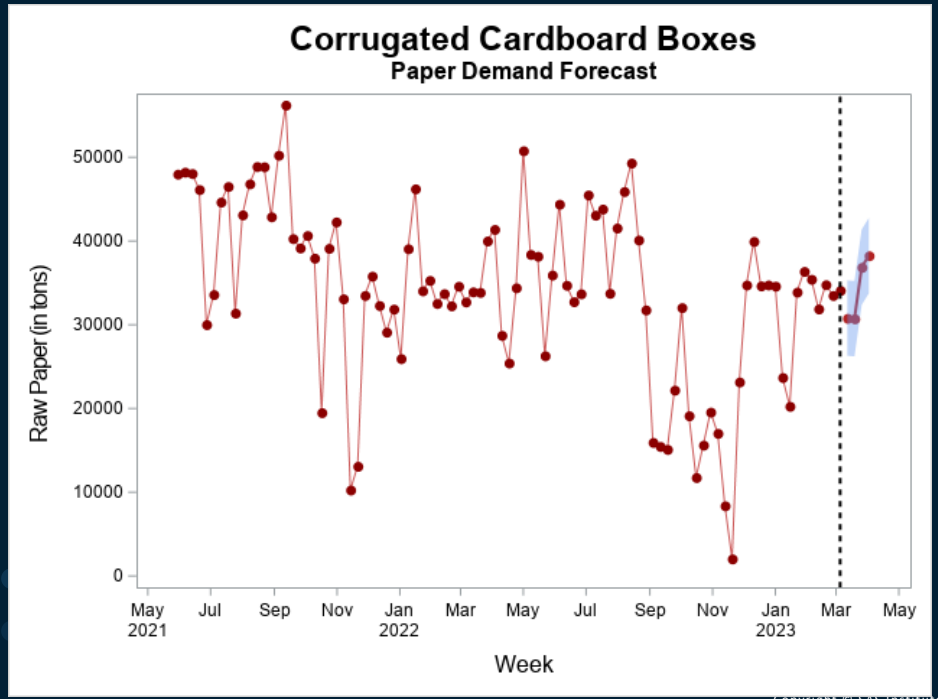
```
1  proc tsmodel data = mylib.atexpert outsum=mylib.tonssum
2               outobj = (deptFor  = mylib.tonsFor
3                         deptEst  = mylib.tonsEst
4                         deptStat = mylib.tonsStat);
5    by box_type;
6    id date interval=week;
7    var paper_demand /acc = total;
8
9    *use ATSM package;
10    require atsm;
11     submit;
12      *declare ATSM objects;
13       declare object dataFrame(tsdf);
14       declare object myDiagnose(diagnose);
15       declare object myDiagSpec(diagspec);
16       declare object forecast(foreng);
17       declare object deptFor(outfor);
18       declare object deptEst(outest);
19       declare object deptStat(outstat);
```

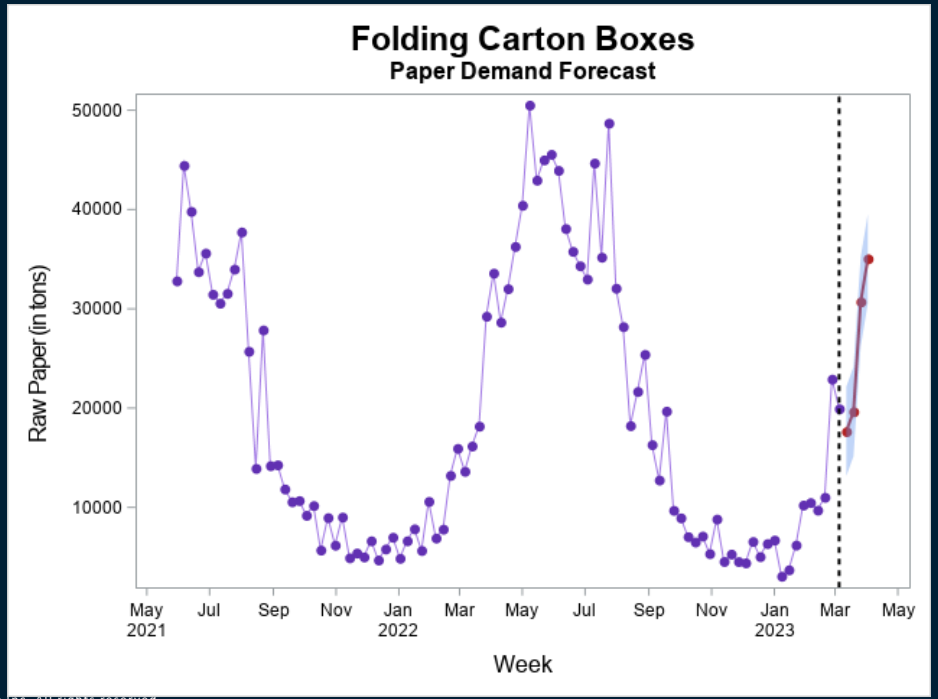
# SAS® Visual Forecasting Screenshots

```
21          *set up dependent and independent variables;
22          rc = dataFrame.initialize();
23          rc = dataFrame.addY(tons);
24
25          *set up time series diagnose specifications;
26          rc = myDiagSpec.open();
27          rc = myDiagSpec.setArimax;
28          rc = myDiagSpec.setEsm('method', 'best');
29          rc = myDiagSpec.close();
30
31          *diagnose time series to generate candidate model list;
32          rc = myDiagnose.initialize(dataFrame);
33          rc = myDiagnose.setSpec(myDiagSpec);
34          rc = myDiagnose.run();
35
36          *run model selection and forecast;
37          rc = forecast.initialize(myDiagnose);
38          rc = forecast.setOption('lead', 4);
39          rc = forecast.run();
40
41          *collect forecast results;
42          rc = deptFor.collect(forecast);
43          rc = deptEst.collect(forecast);
44          rc = deptStat.collect(forecast);
45      endsubmit;
46  run;
```

# Historical Paper Demand By Box Type

| | t+1 | t+2 | t+3 | t+4 |
|---|---|---|---|---|
| Total Raw Paper Forecasts (tons) | 53,089 | 54,047 | 67,436 | 70,515 |



# Business Problem Summary

| | t | t+1 | t+2 | t+3 | t+4 |
|---|---|---|---|---|---|
| Total Raw Paper Forecasts (tons) | - | 53,089 | 54,047 | 67,436 | 70,515 |
| Recycled Paper Plant Production (tons) | - | 12,000 | 18,000 | 20,000 | 22,000 |
| Recycled Paper Plant Cost (per ton) | - | $0.10 | $0.10 | $0.18 | $0.18 |
| Current New Paper Inventory (tons) | 45,000 | - | - | - | - |
| Current Recycled Paper Inventory (tons) | 62,500 | - | - | - | - |

| Tier | Minimum Tons Purchased Per Week | Price Per Ton |
|---|---|---|
| Tier 1 | ≥ 10,000 | $0.15 |
| Tier 2 | ≥ 25,000 | $0.12 |
| Tier 3 | ≥ 35,000 | $0.09 |
| Tier 4 | ≥ 50,000 | $0.07 |

Using this input data, along with the requirement that at least 25% of the paper used each week to make boxes must be recycled:

Which new paper contract tier allows the **Box Manufacturer** to meet forecasted demand at the lowest possible cost?

# Solvers in SAS Optimization

- Linear Programming
- Mixed Integer Linear Programming
- Quadratic Programming
- Conic Optimization
- Nonlinear Programming
- Constraint Programming
- Network Algorithms
- Black-Box Optimization

# PROC OPTMODEL: Major Features

- Algebraic modeling language with optimization-oriented syntax:
  - Variables, constraints, bounds, objectives
  - Algebraic expression of functions
  - Parameters, arrays, index sets
- Close integration with SAS programming environment
- Flexible input/output: read from and create data sets
- Separation between model structure and instance data
- Interactive environment
- Direct access to LP, MILP, QP, conic, NLP, CLP, network, and black-box solvers
- Support for customized algorithms
- Standard and user-defined functions

# Input Data for Optimization

forecast_data

| Obs | week | paper_demand | unit_cost | paper_production |
|---|---|---|---|---|
| 1 | 1 | 53089 | 0.10 | 12000 |
| 2 | 2 | 54047 | 0.10 | 18000 |
| 3 | 3 | 67436 | 0.18 | 20000 |
| 4 | 4 | 70515 | 0.18 | 22000 |

contract_data

| Obs | tier | price | purchase_lb |
|---|---|---|---|
| 1 | Tier1 | 0.15 | 10000 |
| 2 | Tier2 | 0.12 | 25000 |
| 3 | Tier3 | 0.09 | 35000 |
| 4 | Tier4 | 0.07 | 50000 |

# Declare Parameters and Read Data

## OPTMODEL code

```
55  proc optmodel;
56     set <num> WEEKS;
57     num demand{WEEKS};     /* projected demand across all box types */
58     num cost{WEEKS};       /* weekly unit cost for recycled paper */
59     num production{WEEKS}; /* weekly recycled production amount from Recycled Paper Plant */
60     num production_cost{t in WEEKS} = cost[t]*production[t]; /* total weekly production cost for recycled paper */
61
62     set <str> CONTRACTS;
63     num price{CONTRACTS};       /* weekly unit cost for new paper for each contract tier */
64     num purchase_lb{CONTRACTS}; /* lower bound (i.e., minimum) purchase quantity of new paper for each contract tier */
65
66     read data forecast_data into WEEKS=[week] demand=paper_demand cost=unit_cost production=paper_production;
67     read data contract_data into CONTRACTS=[tier] price purchase_lb;
68
69     str contract; /* ith contract tier */
```

# Declare Decision Variables and Objective

## OPTMODEL code

```
71      var Purchase{WEEKS} >= purchase_lb[contract];
72      var UsePurchased{WEEKS} >= 0;
73      var UseRecycled{WEEKS} >= 0;
74      var PurchaseInventory{WEEKS} >= 0;
75      var RecycledInventory{WEEKS} >= 0;
76
77      impvar PurchaseCost{t in WEEKS} = price[contract]*Purchase[t];
78      impvar PaperCost{t in WEEKS} = PurchaseCost[t] + production_cost[t];
79
80      min TotalPaperCost = sum{t in WEEKS} PaperCost[t];
```

# Declare Constraints

## OPTMODEL code

```
82    con MeetDemand{t in WEEKS}:
83       UsePurchased[t] + UseRecycled[t] = demand[t];
84
85    con PurchaseInventoryBalance{t in WEEKS}:
86       PurchaseInventory[t] = (if t = 1 then 45000 else PurchaseInventory[t-1]) + Purchase[t] - UsePurchased[t];
87
88    con RecycledInventoryBalance{t in WEEKS}:
89       RecycledInventory[t] = (if t = 1 then 62500 else RecycledInventory[t-1]) + production[t] - UseRecycled[t];
90
91    con RecycledPercent{t in WEEKS}:
92       UseRecycled[t] >= 0.25*(UsePurchased[t] + UseRecycled[t]);
```

# Call LP Solver in a Loop

## OPTMODEL code

```
103    for {c in CONTRACTS} do;
104       put c=;
105       contract = c;
106       solve;* with lp / algorithm=dual;
107       print {t in WEEKS} (UseRecycled[t]/(UsePurchased[t]+UseRecycled[t]));
108       create data (contract||'_Solution') from [Week]={t in WEEKS} contract Demand Price[contract] Purchase
109          UsePurchased PurchaseInventory Cost Production UseRecycled RecycledInventory PaperCost
110          production_cost PurchaseCost;
111       print contract TotalPaperCost;
112    end;
```

# Call LP Solver in a Loop

```
c=Tier1
NOTE: Problem generation will use 4 threads.
NOTE: The problem has 20 variables (0 free, 0 fixed).
NOTE: The problem uses 8 implicit variables.
NOTE: The problem has 16 linear constraints (0 LE, 12 EQ, 4 GE, 0 range).
NOTE: The problem has 42 linear constraint coefficients.
NOTE: The problem has 0 nonlinear constraints (0 LE, 0 EQ, 0 GE, 0 range).
NOTE: The OPTMODEL presolver is disabled for linear problems.
NOTE: The LP presolver value AUTOMATIC is applied.
NOTE: The LP presolver time is 0.00 seconds.
NOTE: The LP presolver removed 12 variables and 10 constraints.
NOTE: The LP presolver removed 27 constraint coefficients.
NOTE: The presolved problem has 8 variables, 6 constraints, and 15 constraint coefficients.
NOTE: The LP solver is called.
NOTE: The Dual Simplex algorithm is used.
                          Objective
      Phase Iteration        Value         Time
       D 2          1    3.810000E+03         0
       P 2         11    2.039805E+04         0
NOTE: Optimal.
NOTE: Objective = 20398.05.
NOTE: The Dual Simplex solve time is 0.00 seconds.
NOTE: The data set WORK.TIER1_SOLUTION has 4 observations and 14 variables.
c=Tier2
NOTE: Problem generation will use 4 threads.
NOTE: The problem has 20 variables (0 free, 0 fixed).
```

# Optimization Results

The OPTMODEL Procedure

| Problem Summary | |
|---|---|
| Objective Sense | Minimization |
| Objective Function | TotalPaperCost |
| Objective Type | Linear |
| | |
| Number of Variables | 20 |
| Bounded Above | 0 |
| Bounded Below | 20 |
| Bounded Below and Above | 0 |
| Free | 0 |
| Fixed | 0 |
| | |
| Number of Constraints | 16 |
| Linear LE (<=) | 0 |
| Linear EQ (=) | 12 |
| Linear GE (>=) | 4 |
| Linear Range | 0 |
| | |
| Constraint Coefficients | 42 |

| Solution Summary | |
|---|---|
| Solver | LP |
| Algorithm | Dual Simplex |
| Objective Function | TotalPaperCost |
| Solution Status | Optimal |
| Objective Value | 20398.05 |
| | |
| Primal Infeasibility | 0 |
| Dual Infeasibility | 0 |
| Bound Infeasibility | 0 |
| | |
| Iterations | 11 |
| Presolve Time | 0.00 |
| Solution Time | 0.01 |

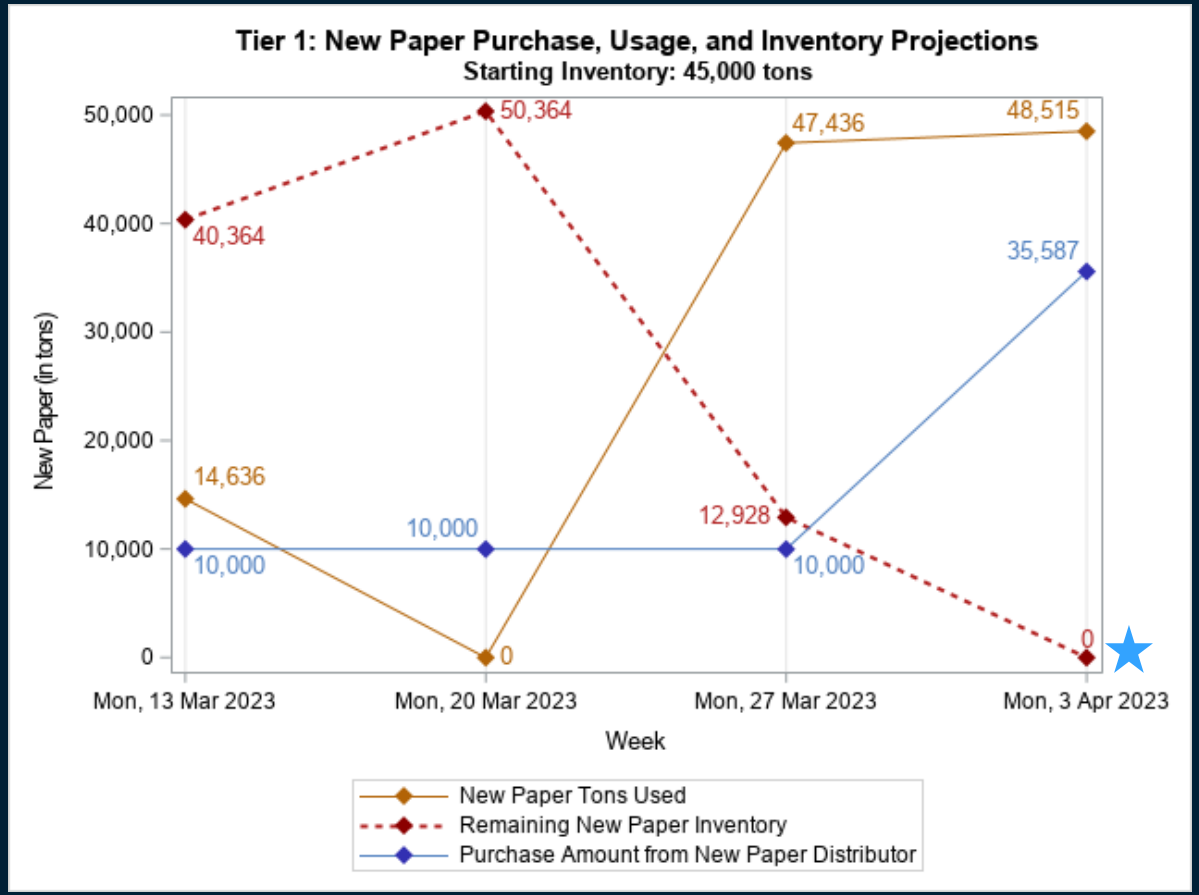
| [1] | |
|---|---|
| 1 | 0.53917 |
| 2 | 0.25000 |
| 3 | 0.47228 |
| 4 | 0.85819 |

| contract | TotalPaperCost |
|---|---|
| Tier1 | 20398 |

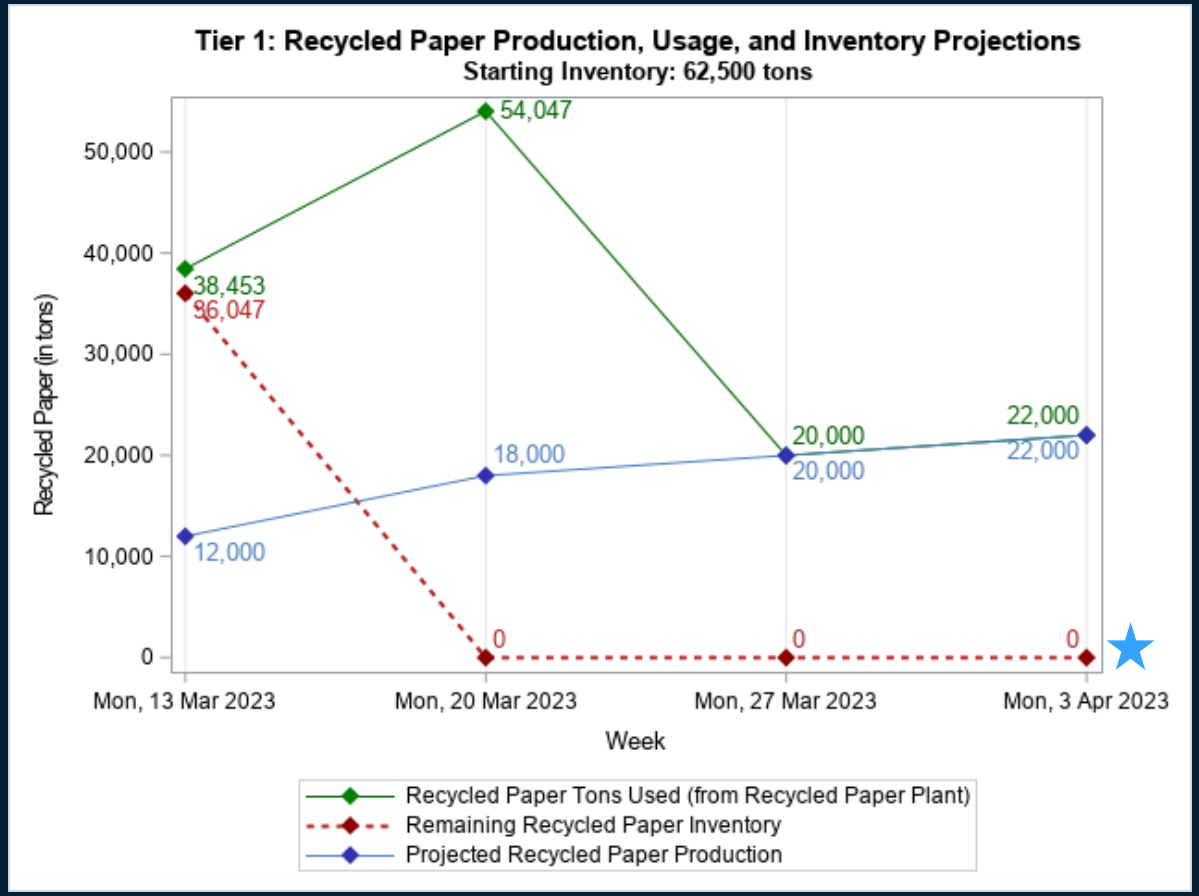
# Optimization Results

| Contract | Total Paper Cost |
|---|---|
| Tier 1* | $20,398* |
| Tier 2 | $22,560 |
| Tier 3 | $23,160 |
| Tier 4 | $24,560 |

# Optimization Results: Tier 1



# Optimization Results: Tier 1



# Modify the Problem

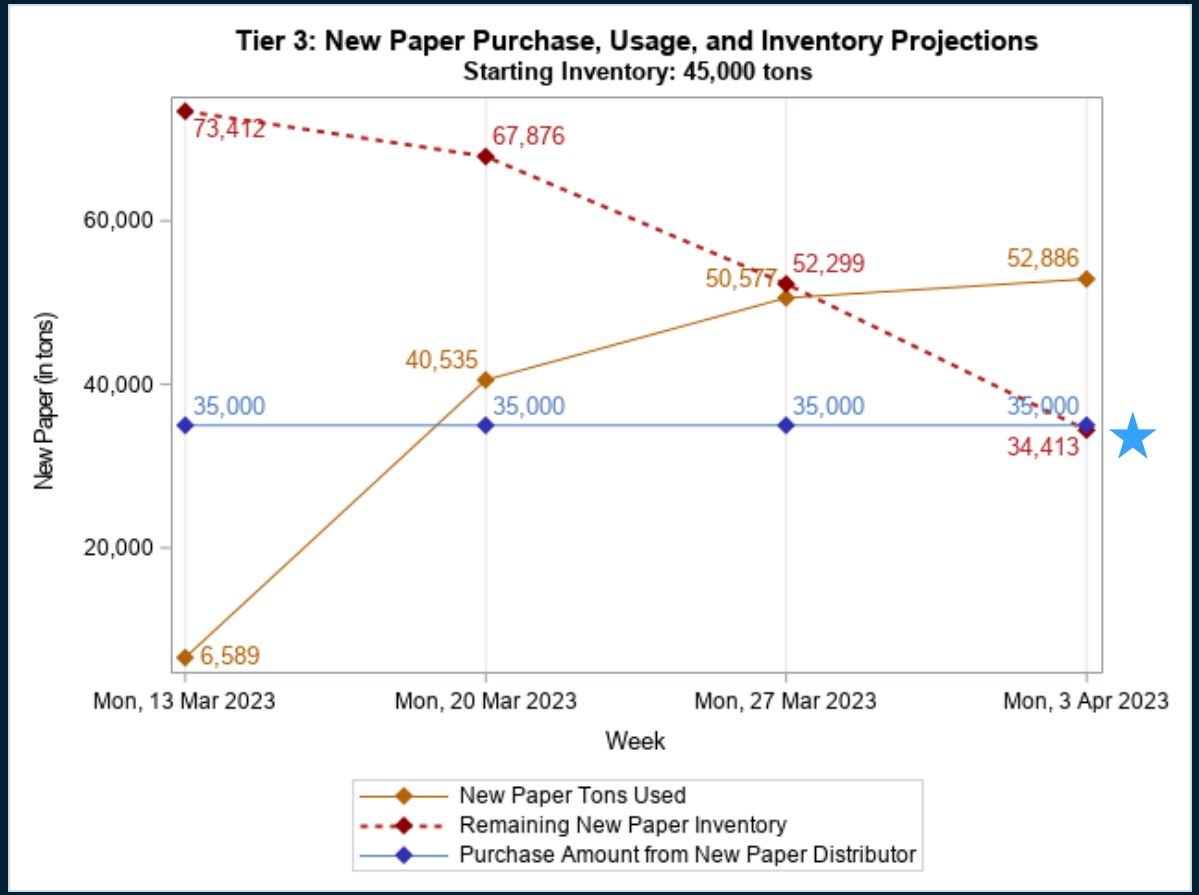
- New business rule: final inventory must be ≥ 30000 tons of new paper and ≥ 40000 tons of recycled paper
- Add explicit constraint or modify the .LB variable suffix

```
 95    /*   con EndingPurchaseInventory:*/
 96    /*      PurchaseInventory[4] >= 30000;*/
 97          PurchaseInventory[4].lb = 30000;
 98
 99    /*   con EndingRecycledInventory:*/
100    /*      RecycledInventory[4] >= 40000;*/
101         RecycledInventory[4].lb = 40000;
```

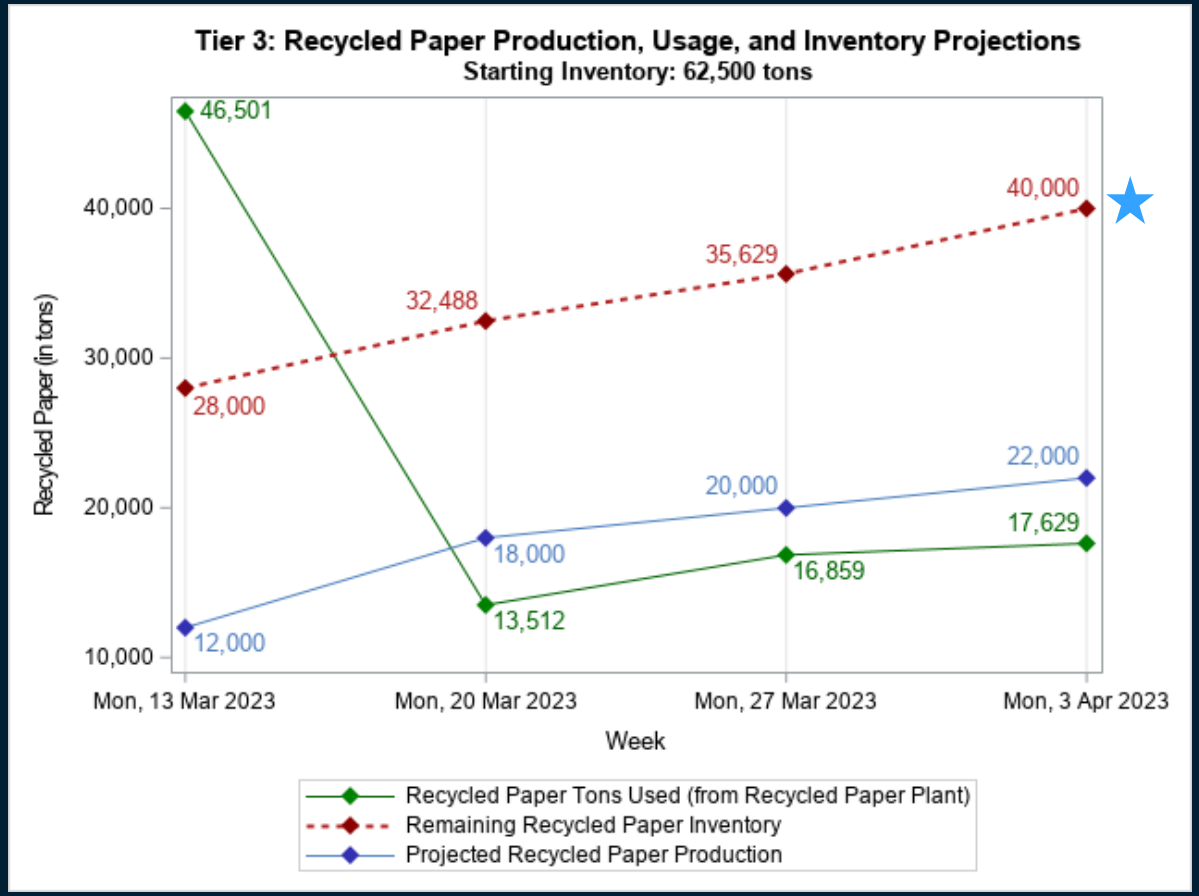
# Optimization Results

| Contract | TotalPaperCost | TotalPaperCost with .lb |
|---|---|---|
| Tier 1 | $20,398 | $30,898 |
| Tier 2 | $22,560 | $26,830 |
| Tier 3* | $23,160 | $23,160* |
| Tier 4 | $24,560 | $24,560 |

# Optimization Results: Tier 3



# Optimization Results: Tier 3



# Use COFOR Loop to Solve Concurrently

## OPTMODEL code

```
103     cofor {c in CONTRACTS} do;
104        put c=;
105        contract = c;
106        solve;* with lp / algorithm=dual;
107        print {t in WEEKS} (UseRecycled[t]/(UsePurchased[t]+UseRecycled[t]));
108        create data (contract||'_Solution') from [Week]={t in WEEKS} contract Demand Price[contract] Purchase
109           UsePurchased PurchaseInventory Cost Production UseRecycled RecycledInventory PaperCost
110           production_cost PurchaseCost;
111        print contract TotalPaperCost;
112     end;
```

# Use COFOR Loop to Solve Concurrently

```
NOTE: The COFOR statement is executing in single-machine mode.
c=Tier1
NOTE: Problem generation will use 4 threads.
NOTE: The problem has 20 variables (0 free, 0 fixed).
NOTE: The problem uses 8 implicit variables.
NOTE: The problem has 16 linear constraints (0 LE, 12 EQ, 4 GE, 0 range).
NOTE: The problem has 42 linear constraint coefficients.
NOTE: The problem has 0 nonlinear constraints (0 LE, 0 EQ, 0 GE, 0 range).
NOTE: The OPTMODEL presolver is disabled for linear problems.
NOTE: The LP presolver value AUTOMATIC is applied.
NOTE: The LP presolver time is 0.00 seconds.
NOTE: The LP presolver removed 18 variables and 15 constraints.
NOTE: The LP presolver removed 40 constraint coefficients.
NOTE: The presolved problem has 2 variables, 1 constraints, and 2 constraint coefficients.
NOTE: The LP solver is called.
NOTE: The Network Simplex algorithm is used.
NOTE: The Network Simplex algorithm was not run because the number of constraints (after presolve) is less than or equal to 1.
NOTE: The Dual Simplex algorithm is used.
                          Objective
      Phase Iteration        Value         Time
       D 2          1    3.089805E+04         0
NOTE: Optimal.
NOTE: Objective = 30898.05.
NOTE: The Simplex solve time is 0.00 seconds.
NOTE: The data set WORK.TIER1_SOLUTION has 4 observations and 14 variables.
c=Tier2
NOTE: Problem generation will use 3 threads.
NOTE: The problem has 20 variables (0 free, 0 fixed).
```

# Input Data for Optimization

forecast_data

| Obs | week | paper_demand | unit_cost | paper_production |
|---|---|---|---|---|
| 1 | 1 | 53089 | 0.10 | 12000 |
| 2 | 2 | 54047 | 0.10 | 18000 |
| 3 | 3 | 67436 | 0.18 | 20000 |
| 4 | 4 | 70515 | 0.18 | 22000 |

contract_data

| Obs | tier | price | purchase_lb |
|---|---|---|---|
| 1 | Tier1 | 0.15 | 10000 |
| 2 | Tier2 | 0.12 | 25000 |
| 3 | Tier3 | 0.09 | 35000 |
| 4 | Tier4 | 0.07 | 50000 |

# Use runOptmodel groupBy to Solve Concurrently

```
57  proc cas noqueue;
58     source pgm;
59     put _BY_LINE_;
60     set <num> WEEKS;
61     num demand{WEEKS};      /* projected demand across all box types */
62     num cost{WEEKS};        /* weekly unit cost for recycled paper */
63     num production{WEEKS}; /* weekly recycled production amount from Recycled Paper Plant */
64     num production_cost{t in WEEKS} = cost[t]*production[t]; /* total weekly production cost for recycled paper */
65
66     num price;          /* weekly unit cost for new paper for each contract tier */
67     num purchase_lb; /* lower bound (i.e., minimum) purchase quantity of new paper for each contract tier */
68
69     read data forecast_data nogroupby into WEEKS=[week] demand=paper_demand cost=unit_cost production=paper_production;
70     read data contract_data into price purchase_lb;
71
72     var Purchase{WEEKS} >= purchase_lb;
73     var UsePurchased{WEEKS} >= 0;
74     var UseRecycled{WEEKS} >= 0;
```

# Use runOptmodel groupBy to Solve Concurrently

```
98
99      solve;* with lp / algorithm=dual;
100     print {t in WEEKS} (UseRecycled[t]/(UsePurchased[t]+UseRecycled[t]));
101     create data Solution from [Week]={t in WEEKS} Demand Price Purchase
102        UsePurchased PurchaseInventory Cost Production UseRecycled RecycledInventory PaperCost
103        production_cost PurchaseCost;
104     print TotalPaperCost;
105     endsource;
106
107     /* solve a separate problem for each contract tier */
108     action optimization.runOptmodel / code=pgm groupBy='tier' statusOut={name='statusOut',replace=true};
109     run;
110  quit;
```

# Use runOptmodel groupBy to Solve Concurrently

```
tier=Tier1
NOTE: There were 4 rows read from table 'FORECAST_DATA' in caslib 'CASUSER(robpra)'.
NOTE: There were 1 rows read from table 'CONTRACT_DATA' in caslib 'CASUSER(robpra)'.
NOTE: Problem generation will use 8 threads.
NOTE: The problem has 20 variables (0 free, 0 fixed).
NOTE: The problem uses 8 implicit variables.
NOTE: The problem has 16 linear constraints (0 LE, 12 EQ, 4 GE, 0 range).
NOTE: The problem has 42 linear constraint coefficients.
NOTE: The problem has 0 nonlinear constraints (0 LE, 0 EQ, 0 GE, 0 range).
NOTE: The OPTMODEL presolver is disabled for linear problems.
NOTE: The LP presolver value AUTOMATIC is applied.
NOTE: The LP presolver time is 0.00 seconds.
NOTE: The LP presolver removed 18 variables and 15 constraints.
NOTE: The LP presolver removed 40 constraint coefficients.
NOTE: The presolved problem has 2 variables, 1 constraints, and 2 constraint coefficients.
NOTE: The LP solver is called.
NOTE: The Network Simplex algorithm is used.
NOTE: The Network Simplex algorithm was not run because the number of constraints (after presolve) is less than or equal to 1.
NOTE: The Dual Simplex algorithm is used.
                            Objective
      Phase Iteration         Value         Time
       D 2          1    3.089805E+04          0
NOTE: Optimal.
NOTE: Objective = 30898.05.
NOTE: The Simplex solve time is 0.00 seconds.
NOTE: The output table 'SOLUTION' in caslib 'CASUSER(robpra)' added 4 rows for the current BY group.
tier=Tier2
NOTE: There were 4 rows read from table 'FORECAST_DATA' in caslib 'CASUSER(robpra)'.
NOTE: There were 1 rows read from table 'CONTRACT_DATA' in caslib 'CASUSER(robpra)'.
NOTE: Problem generation will use 8 threads.
```

# Use runOptmodel groupBy to Solve Concurrently

## Solution Table

| Obs | tier | Week | demand | price | Purchase | UsePurchased | PurchaseInventory | cost | production | UseRecycled | RecycledInventory | PaperCost | production_cost | PurchaseCost |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Tier1 | 2 | 54047 | 0.15 | 105587 | 40535.25 | 113463.25 | 0.10 | 18000 | 13511.75 | 32487.75 | 17638.05 | 1800 | 15838.05 |
| 2 | Tier1 | 3 | 67436 | 0.15 | 10000 | 50577.00 | 72886.25 | 0.18 | 20000 | 16859.00 | 35628.75 | 5100.00 | 3600 | 1500.00 |
| 3 | Tier1 | 4 | 70515 | 0.15 | 10000 | 52886.25 | 30000.00 | 0.18 | 22000 | 17628.75 | 40000.00 | 5460.00 | 3960 | 1500.00 |
| 4 | Tier1 | 1 | 53089 | 0.15 | 10000 | 6588.50 | 48411.50 | 0.10 | 12000 | 46500.50 | 27999.50 | 2700.00 | 1200 | 1500.00 |
| 5 | Tier2 | 2 | 54047 | 0.12 | 25000 | 40535.25 | 47876.25 | 0.10 | 18000 | 13511.75 | 32487.75 | 4800.00 | 1800 | 3000.00 |
| 6 | Tier2 | 3 | 67436 | 0.12 | 60587 | 50577.00 | 57886.25 | 0.18 | 20000 | 16859.00 | 35628.75 | 10870.44 | 3600 | 7270.44 |
| 7 | Tier2 | 4 | 70515 | 0.12 | 25000 | 52886.25 | 30000.00 | 0.18 | 22000 | 17628.75 | 40000.00 | 6960.00 | 3960 | 3000.00 |
| 8 | Tier2 | 1 | 53089 | 0.12 | 25000 | 6588.50 | 63411.50 | 0.10 | 12000 | 46500.50 | 27999.50 | 4200.00 | 1200 | 3000.00 |
| 9 | Tier3 | 2 | 54047 | 0.09 | 35000 | 40535.25 | 67876.25 | 0.10 | 18000 | 13511.75 | 32487.75 | 4950.00 | 1800 | 3150.00 |
| 10 | Tier3 | 3 | 67436 | 0.09 | 35000 | 50577.00 | 52299.25 | 0.18 | 20000 | 16859.00 | 35628.75 | 6750.00 | 3600 | 3150.00 |
| 11 | Tier3 | 4 | 70515 | 0.09 | 35000 | 52886.25 | 34413.00 | 0.18 | 22000 | 17628.75 | 40000.00 | 7110.00 | 3960 | 3150.00 |
| 12 | Tier3 | 1 | 53089 | 0.09 | 35000 | 6588.50 | 73411.50 | 0.10 | 12000 | 46500.50 | 27999.50 | 4350.00 | 1200 | 3150.00 |
| 13 | Tier4 | 2 | 54047 | 0.07 | 50000 | 40535.25 | 64648.00 | 0.10 | 18000 | 13511.75 | 65716.00 | 5300.00 | 1800 | 3500.00 |
| 14 | Tier4 | 3 | 67436 | 0.07 | 50000 | 50577.00 | 64071.00 | 0.18 | 20000 | 16859.00 | 68857.00 | 7100.00 | 3600 | 3500.00 |
| 15 | Tier4 | 4 | 70515 | 0.07 | 50000 | 52886.25 | 61184.75 | 0.18 | 22000 | 17628.75 | 73228.25 | 7460.00 | 3960 | 3500.00 |
| 16 | Tier4 | 1 | 53089 | 0.07 | 50000 | 39816.75 | 55183.25 | 0.10 | 12000 | 13272.25 | 61227.75 | 4700.00 | 1200 | 3500.00 |

# Use runOptmodel groupBy to Solve Concurrently

## statusOut Table

| Obs | tier | status | algorithm | solutionStatus | objective | numIterations | presolveTime | solutionTime | numIterations2 | primalInf | dualInf | boundInf |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | Tier1 | OK | NS | OPTIMAL | 30898.05 | 0 | .004929066 | .005860806 | 1 | 1.819E-16 | 0 | 0 |
| 2 | Tier2 | OK | NS | OPTIMAL | 26830.44 | 0 | .000683069 | .001243830 | 1 | 1.819E-16 | 0 | 0 |
| 3 | Tier3 | OK | NS | OPTIMAL | 23160.00 | 0 | .000551939 | .001891136 | 2 | 1.819E-16 | 0 | 0 |
| 4 | Tier4 | OK | SPX | OPTIMAL | 24560.00 | 0 | .000469208 | .000495195 | . | 0 | 0 | 0 |

# Indicator Constraints

- New business rule: purchase 0 or at least purchase_lb[contract]
- Use binary decision variables and indicator constraints

```
19      /*   var Purchase{WEEKS} >= purchase_lb[contract]; */
20         var Purchase{t in WEEKS} >= 0 <= demand[t];
```

```
48      var IfPurchase{WEEKS} binary;
49      con Indicator0{t in WEEKS}:
50         IfPurchase[t] = 0 implies Purchase[t] = 0;
51      con Indicator1{t in WEEKS}:
52         IfPurchase[t] = 1 implies Purchase[t] >= purchase_lb[contract];
```

# Indicator Constraints

- Expand original and automatically linearized constraints
- Call MILP solver

```
53    cofor {c in CONTRACTS} do;
54       put c=;
55       contract = c;
56       expand Indicator0;
57       expand Indicator1;
58       expand Indicator0 / linearize;
59       expand Indicator1 / linearize;
60       solve;* with milp;
61       print {t in WEEKS} (UseRecycled[t]/(UsePurchased[t]+UseRecycled[t]));
62       create data (contract||'_Solution') from [Week]={t in WEEKS} contract Demand Price[contract] Purchase
63          UsePurchased PurchaseInventory Cost Production UseRecycled RecycledInventory PaperCost
64          production_cost PurchaseCost;
65       print contract TotalPaperCost;
66    end;
```

# Indicator Constraints

## Output from EXPAND Statements

EXPAND

```
Constraint Indicator0[1]: IfPurchase[1] = 0 IMPLIES Purchase[1] = 0
Constraint Indicator0[2]: IfPurchase[2] = 0 IMPLIES Purchase[2] = 0
Constraint Indicator0[3]: IfPurchase[3] = 0 IMPLIES Purchase[3] = 0
Constraint Indicator0[4]: IfPurchase[4] = 0 IMPLIES Purchase[4] = 0

Constraint Indicator1[1]: IfPurchase[1] = 1 IMPLIES Purchase[1] >= 10000
Constraint Indicator1[2]: IfPurchase[2] = 1 IMPLIES Purchase[2] >= 10000
Constraint Indicator1[3]: IfPurchase[3] = 1 IMPLIES Purchase[3] >= 10000
Constraint Indicator1[4]: IfPurchase[4] = 1 IMPLIES Purchase[4] >= 10000
```

EXPAND / LINEARIZE

```
Constraint Indicator0[1]: Purchase[1] - 53089*IfPurchase[1] <= 0
Constraint Indicator0[2]: Purchase[2] - 54047*IfPurchase[2] <= 0
Constraint Indicator0[3]: Purchase[3] - 67436*IfPurchase[3] <= 0
Constraint Indicator0[4]: Purchase[4] - 70515*IfPurchase[4] <= 0

Constraint Indicator1[1]: Purchase[1] - 10000*IfPurchase[1] >= 0
Constraint Indicator1[2]: Purchase[2] - 10000*IfPurchase[2] >= 0
Constraint Indicator1[3]: Purchase[3] - 10000*IfPurchase[3] >= 0
Constraint Indicator1[4]: Purchase[4] - 10000*IfPurchase[4] >= 0
```

# Optimization Results

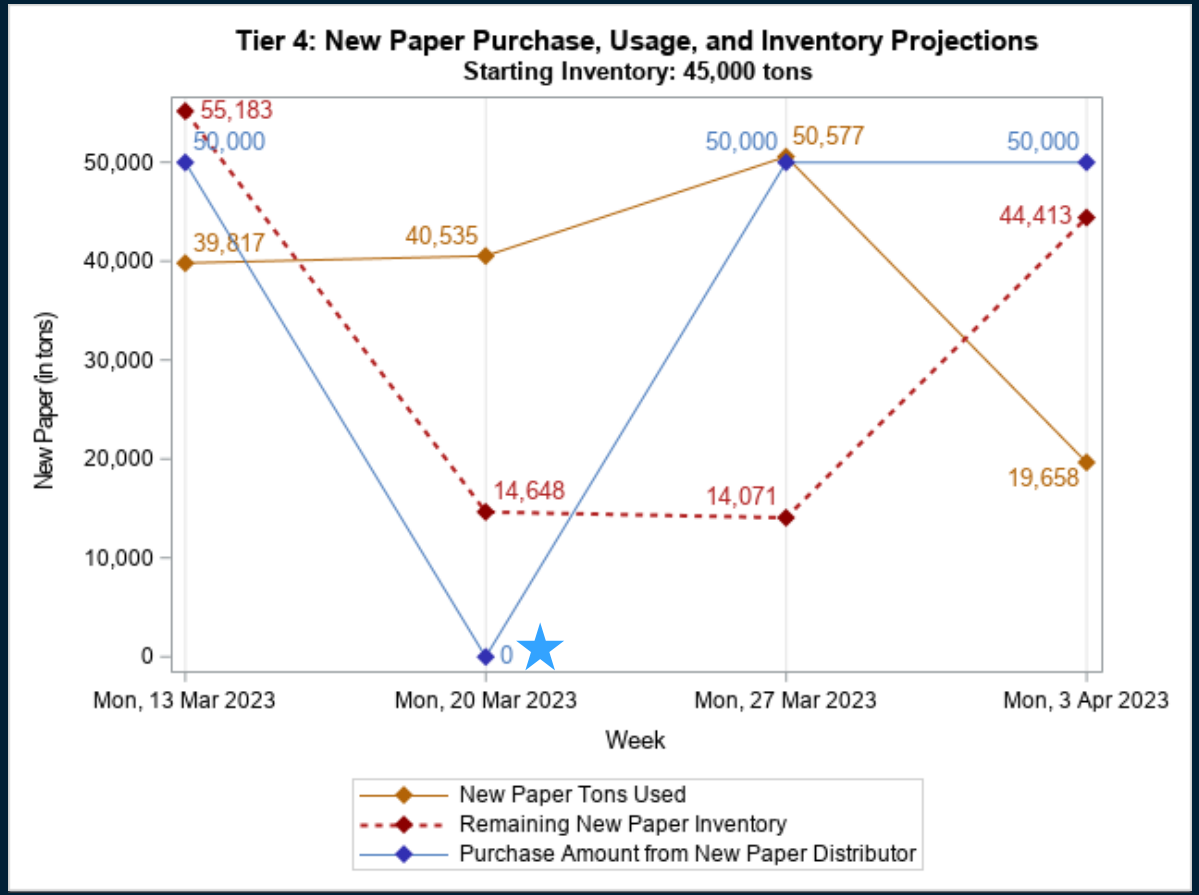
| Problem Summary | |
|---|---|
| Objective Sense | Minimization |
| Objective Function | TotalPaperCost |
| Objective Type | Linear |
| | |
| Number of Variables | 24 |
| Bounded Above | 0 |
| Bounded Below | 16 |
| Bounded Below and Above | 8 |
| Free | 0 |
| Fixed | 0 |
| Binary | 4 |
| Integer | 0 |
| | |
| Number of Constraints | 24 |
| Linear LE (<=) | 0 |
| Linear EQ (=) | 16 |
| Linear GE (>=) | 8 |
| Linear Range | 0 |
| | |
| Constraint Coefficients | 50 |

| Solution Summary | |
|---|---|
| Solver | MILP |
| Algorithm | Branch and Cut |
| Objective Function | TotalPaperCost |
| Solution Status | Optimal |
| Objective Value | 30898.05 |
| | |
| Relative Gap | 0 |
| Absolute Gap | 0 |
| Primal Infeasibility | 0 |
| Bound Infeasibility | 0 |
| Integer Infeasibility | 0 |
| | |
| Best Bound | 30898.05 |
| Nodes | 1 |
| Solutions Found | 1 |
| Iterations | 7 |
| Presolve Time | 0.00 |
| Solution Time | 0.01 |

# Optimization Results

| Contract | TotalPaperCost | TotalPaperCost with .lb | TotalPaperCost with .lb and Indicator Cons |
|---|---|---|---|
| Tier 1 | $20,398 | $30,898 | $30,898 |
| Tier 2 | $22,560 | $26,830 | $26,830 |
| Tier 3 | $23,160 | $23,160 | $22,763 |
| Tier 4* | $24,560 | $24,560 | $21,060* |

# Optimization Results: Tier 4



# Summary

- Forecasting and optimization play well together
- SAS Visual Forecasting automatically generates, manages, and deploys large numbers of trustworthy forecasts
  - Champion models are chosen from time series, ML, hybrid (ML + time series), and deep learning forecasting algorithms
  - Output forecast tables are effortlessly read into SAS Optimization procedures
- SAS Optimization contains modeling language, broad range of solvers
  - IF-THEN/ELSE expression
  - FOR loop, COFOR loop, runOptmodel groupBy
  - Indicator constraints, automated linearization
  - EXPAND original or linearized model