



Recherche Opérationnelle

Rapport TP1

2SN L4

Élèves :

THEVENET Louis

SABLAYROLLES Guillaume

9 Décembre 2024

Table des matières

1	Modélisation et Résolution de PL/PLNE avec le solveur GLPK	3
1.1	Assemblage	3
1.2	Applications en optimisation pour l'e-commerce	5

1 Modélisation et Résolution de PL/PLNE avec le solveur GLPK

1.1 Assemblage

Ce problème peut se modéliser par PL dans le cas où la fabrication interrompue en fin de semaine d'un vélo peut être continuée en début de la semaine suivante. Au contraire, si on est obligé de fabriquer les nouveaux vélos de zéro chaque semaine, le problème se modélise par PLNE.

Variables

Nombre de vélos cargos $C \in \mathbb{R}^+$ (ou entière dans le cas PLNE)

Nombre de vélos cargos $S \in \mathbb{R}^+$ (ou entière dans le cas PLNE)

Fonction objectif

$$f(C, S) = \max(700C + 300S)$$

Contraintes

Respect du nombre d'heures $0.06C + 0.05S \leq 60$

Respect de la surface maximale occupée $2.5C + 1S \leq 1500$

Respect du nombre max de vélos cargos produits $C \leq 700$

Solution PLNE

```
1 Problem:
2 Rows:      3
3 Columns:    2 (2 integer, 0 binary)
4 Non-zeros:  5
5 Status:     INTEGER OPTIMAL
6 Objective:  Benefice = 438400 (MAXimum)
7
8   No.  Row name      Activity  Lower bound  Upper bound
9   ----  -
10      1 TravailHebdo      59.92              60
11      2 SurfaceOccupee      1500             1500
12      3 ProductionCargoMax      232              700
13
14   No. Column name      Activity  Lower bound  Upper bound
15   ----  -
16      1 C      *      232              0
17      2 S      *      920              0
18
19
20
21 Integer feasibility conditions:
22
23 KKT.PE: max.abs.err = 0.00e+00 on row 0
24         max.rel.err = 0.00e+00 on row 0
25         High quality
26
27 KKT.PB: max.abs.err = 0.00e+00 on row 0
28         max.rel.err = 0.00e+00 on row 0
29         High quality
30
31 End of output
```

On constate que la solution trouvée $(C, S) = (232, 920)$ maximise l'objectif avec $f(C, S) = 438400\text{€}$. Le nombre d'heures nécessaires pour ce résultat est 59.92h et la surface disponible

est complètement utilisée. Si on augmente la surface disponible, on peut alors produire plus de vélos, on peut également faire varier le ratio $\frac{\text{Place occupée par un vélo cargo}}{\text{Place occupée par un vélo normal}}$, ce qui permettrait de produire plus de vélos cargo (la limite de 700 n'est pas atteinte car ce n'est pas « rentable » de faire des cargos avec ces paramètres.).

Ce problème se modélise facilement sous forme « .lp » car les contraintes sont explicites et qu'il n'y a pas beaucoup de variables à contraindre.

1.1.1 Affectation avec prise en compte des préférences

Données

$n \in \mathbb{N}$ nombre de personnes

$m \in \mathbb{N}$ nombre d'activités

$P \in \mathcal{M}_{n,m}(\mathbb{R})$ Matrice des préférences

Variables

On utilise une matrice $M \in \mathcal{M}_{n,m}(\mathbb{R})$ telle que

$$\forall 1 \leq i \leq n, \forall 1 \leq j \leq m, M_{i,j} = \begin{cases} 1 & \text{si la personne } i \text{ réalise l'activité } j \\ 0 & \text{sinon} \end{cases}$$

Fonction objectif

$$f : \begin{cases} \mathcal{M}_{n,m}(\mathbb{R}) \rightarrow \mathbb{R} \\ M \mapsto \max \left(\sum_{i=1}^n \sum_{j=1}^m M_{i,j} \times P_{i,j} \right) \end{cases}$$

Contraintes

Une personne est associée à une seule activité $\forall 1 \leq i \leq m, \sum_{j=1}^m M_{i,j} = 0$

Une activité est associée à une seule personne $\forall 1 \leq j \leq m, \sum_{i=1}^n M_{i,j} = 0$

Solution

Pour $P = \begin{pmatrix} 9 & 5 & 1 \\ 2 & 4 & 2 \\ 9 & 4 & 8 \end{pmatrix}$, la solution trouvée est $M = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$. On vérifie aisément que c'est la solution optimale.

```

1 Problem:      PbPreferences
2 Rows:        7
3 Columns:     9 (9 integer, 9 binary)
4 Non-zeros:   27
5 Status:      INTEGER OPTIMAL
6 Objective:    SatisfactionTotale = 21 (MAXimum)

```

No.	Row name	Activity	Lower bound	Upper bound
1	RespectDistributionLigne[P1]			
		1	1	=
2	RespectDistributionLigne[P2]			
		1	1	=
3	RespectDistributionLigne[P3]			
		1	1	=
4	RespectDistributionColonne[T1]			
		1	1	=
5	RespectDistributionColonne[T2]			

```

19          1          1          =
20      6 RespectDistributionColonne[T3]
21          1          1          =
22      7 SatisfactionTotale
23          21
24
25      No. Column name      Activity      Lower bound      Upper bound
26      -----
27      1 M[P1,T1]      *          1          0          1
28      2 M[P1,T2]      *          0          0          1
29      3 M[P1,T3]      *          0          0          1
30      4 M[P2,T1]      *          0          0          1
31      5 M[P2,T2]      *          1          0          1
32      6 M[P2,T3]      *          0          0          1
33      7 M[P3,T1]      *          0          0          1
34      8 M[P3,T2]      *          0          0          1
35      9 M[P3,T3]      *          1          0          1
36
37      Integer feasibility conditions:
38
39      KKT.PE: max.abs.err = 0.00e+00 on row 0
40              max.rel.err = 0.00e+00 on row 0
41              High quality
42
43      KKT.PB: max.abs.err = 0.00e+00 on row 0
44              max.rel.err = 0.00e+00 on row 0
45              High quality
46
47      End of output

```

Nous avons modélisé ce problème sous format « .mod » puisque nous devons faire des sommes sur des éléments de matrices. Le format GMPL offre une plus grande liberté d'expression des contraintes.

C'est le format que nous utiliserons dans la suite.

1.2 Applications en optimisation pour l'e-commerce

1.2.1 Cas particulier 1.1

Données

$f \in \mathbb{N}$ nombre de fluides

$m \in \mathbb{N}$ nombre de magasins

$d \in \mathbb{N}$ nombre de demandes

Et trois matrices:

- fluides_par_demandes $\in \mathcal{M}_{d,f}(\mathbb{R})$
- stock_par_magasin $\in \mathcal{M}_{m,f}(\mathbb{R})$
- cout_par_magasin $\in \mathcal{M}_{m,f}(\mathbb{R})$

Variables

On utilise une matrice $D \in \mathcal{M}_{f,m,d}(\mathbb{R})$ avec

- f le nombre de fluides différents
- m le nombre de magasins
- d le nombre de demandes réalisées telle que

$$\forall 1 \leq i \leq f, \forall 1 \leq j \leq m, \forall 1 \leq k \leq d,$$

$D_{i,j,k}$ est la quantité de fluide i demandée au magasin j lors de la demande k

Fonction objectif

$$f : \begin{cases} \mathcal{M}_{f,m,d}(\mathbb{R}) \rightarrow \mathbb{R} \\ D \mapsto \min \left(\sum_{i=1}^f \sum_{j=1}^m \sum_{k=1}^d \text{cout_par_magasin}_{j,i} \times D_{i,j,k} \right) \end{cases}$$

Contraintes

Pour un fluide et un magasin donnés, la demande totale respecte le stock disponible

$$\forall 1 \leq i \leq f, \forall 1 \leq j \leq m \sum_{k=1}^d D_{i,j,k} \leq \text{stock_par_magasin}_{j,i}$$

Les demandes sont respectées

$$\forall 1 \leq i \leq f, \forall 1 \leq k \leq d \sum_{j=1}^m D_{i,j,k} = \text{fluides_par_demandes}_{k,i}$$

Solution

Pour

$$\text{fluides_par_demandes} = \begin{pmatrix} 2 & 0 \\ 1 & 3 \end{pmatrix},$$

$$\text{stock_par_magasin} = \begin{pmatrix} 2.5 & 1 \\ 1 & 2 \\ 2 & 1 \end{pmatrix},$$

$$\text{cout_par_magasin} = \begin{pmatrix} 1 & 1 \\ 2 & 3 \\ 3 & 2 \end{pmatrix}$$

La solution pour un coût minimum de : CoutTotal = 9.5€ est la matrice $D = [D1, D2]$ avec $D1 = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$ et $D2 = \begin{pmatrix} 0.5 & 0.5 & 0 \\ 1 & 1 & 1 \end{pmatrix}$

1	Problem:	PbMagasin					
2	Rows:	11					
3	Columns:	12					
4	Non-zeros:	36					
5	Status:	OPTIMAL					
6	Objective:	CoutTotal = 9.5 (MINimum)					
7							
8	No.	Row name	St	Activity	Lower bound	Upper bound	Marginal
9	<hr/>						
10	1	RespectStock[F1,M1]					
11		NU		2.5		2.5	-1
12	2	RespectStock[F1,M2]					
13		B		0.5		1	
14	3	RespectStock[F1,M3]					
15		B		0		2	
16	4	RespectStock[F2,M1]					
17		NU		1		1	-2
18	5	RespectStock[F2,M2]					

19			B		1		2	
20	6	RespectStock	[F2,M3]					
21			NU		1		1	-1
22	7	RespectDemande	[F1,D1]					
23			NS		2	2	=	2
24	8	RespectDemande	[F1,D2]					
25			NS		1	1	=	2
26	9	RespectDemande	[F2,D1]					
27			B		0	-0	=	
28	10	RespectDemande	[F2,D2]					
29			NS		3	3	=	3
30	11	CoutTotal	B		9.5			
31								
32	No.	Column	name	St	Activity	Lower bound	Upper bound	Marginal
33								
34	1	D	[F1,M1,D1]	B	2	0		
35	2	D	[F1,M1,D2]	B	0.5	0		
36	3	D	[F1,M2,D1]	NL	0	0		< eps
37	4	D	[F1,M2,D2]	B	0.5	0		
38	5	D	[F1,M3,D1]	NL	0	0		1
39	6	D	[F1,M3,D2]	NL	0	0		1
40	7	D	[F2,M1,D1]	NL	0	0		3
41	8	D	[F2,M1,D2]	B	1	0		
42	9	D	[F2,M2,D1]	NL	0	0		3
43	10	D	[F2,M2,D2]	B	1	0		
44	11	D	[F2,M3,D1]	NL	0	0		3
45	12	D	[F2,M3,D2]	B	1	0		
46								
47	Karush-Kuhn-Tucker optimality conditions:							
48								
49	KKT.PE:	max.abs.err = 0.00e+00 on row 0						
50		max.rel.err = 0.00e+00 on row 0						
51		High quality						
52								
53	KKT.PB:	max.abs.err = 0.00e+00 on row 0						
54		max.rel.err = 0.00e+00 on row 0						
55		High quality						
56								
57	KKT.DE:	max.abs.err = 0.00e+00 on column 0						
58		max.rel.err = 0.00e+00 on column 0						
59		High quality						
60								
61	KKT.DB:	max.abs.err = 0.00e+00 on row 0						
62		max.rel.err = 0.00e+00 on row 0						
63		High quality						
64								
65	End of output							

1.2.2 Cas particulier 1.2

Données

$f \in \mathbb{N}$ nombre de fluides

$m \in \mathbb{N}$ nombre de magasins

$d \in \mathbb{N}$ nombre de demandes

Et cinq matrices:

- fluides_par_demandes $\in \mathcal{M}_{d,f}(\mathbb{R})$
- stock_par_magasin $\in \mathcal{M}_{m,f}(\mathbb{R})$
- cout_par_magasin $\in \mathcal{M}_{m,f}(\mathbb{R})$
- cout_fixe $\in \mathcal{M}_{d,m}(\mathbb{R})$

- $\text{cout_variable} \in \mathcal{M}_{d,m}(\mathbb{R})$

Variables

On utilise une matrice $D \in \mathcal{M}_{f,m,d}(\mathbb{R})$ avec

- f le nombre de fluides différents
- m le nombre de magasins
- d le nombre de demandes réalisées telle que

$$\forall 1 \leq i \leq f \forall 1 \leq j \leq m \forall 1 \leq k \leq d,$$

$D_{i,j,k}$ est la quantité de fluide i demandée au magasin j lors de la demande k

Fonction objectif

$$f : \begin{cases} \mathcal{M}_{f,m,d}(\mathbb{R}) \rightarrow \mathbb{R} \\ D \mapsto \min \left(\sum_{i=1}^f \sum_{j=1}^m \sum_{k=1}^d (\text{cout_par_magasin}_{j,i} + \text{cout_variable}_{k,j}) D_{i,j,k} + \text{cout_fixe}_{k,j} \right) \end{cases}$$

Contraintes

Pour un fluide et un magasin donnés, la demande totale respecte le stock disponible

$$\forall 1 \leq i \leq f, \forall 1 \leq j \leq m \sum_{k=1}^d D_{i,j,k} \leq \text{stock_par_magasin}_{j,i}$$

Les demandes sont respectées

$$\forall 1 \leq i \leq f, \forall 1 \leq k \leq d \sum_{j=1}^m D_{i,j,k} = \text{fluides_par_demandes}_{k,i}$$

Solution

Pour

$$\text{fluides_par_demandes} = \begin{pmatrix} 2 & 0 \\ 1 & 3 \end{pmatrix},$$

$$\text{stock_par_magasin} = \begin{pmatrix} 2.5 & 1 \\ 1 & 2 \\ 2 & 1 \end{pmatrix},$$

$$\text{cout_par_magasin} = \begin{pmatrix} 1 & 1 \\ 2 & 3 \\ 3 & 2 \end{pmatrix},$$

$$\text{cout_fixe} = \begin{pmatrix} 110 & 90 & 100 \\ 110 & 90 & 100 \end{pmatrix},$$

$$\text{cout_variable} = \begin{pmatrix} 10 & 1 & 5 \\ 2 & 20 & 10 \end{pmatrix}$$

La solution pour un coût minimum de $\text{CoutTotal} = 1252\text{€}$ est la matrice $D = [D1, D2]$ avec $D1 = \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 0 \end{pmatrix}$ et $D2 = \begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \end{pmatrix}$

1 **Problem:** Pb2
2 **Rows:** 11


```

3 Columns:      12
4 Non-zeros:   36
5 Status:      OPTIMAL
6 Objective:   CoutTotal = 1252 (MINimum)

```

No.	Row name	St	Activity	Lower bound	Upper bound	Marginal
<hr/>						
1	RespectStock[F1,M1]	B	1		2.5	
2	RespectStock[F1,M2]	NU	1		1	-5
3	RespectStock[F1,M3]	B	1		2	
4	RespectStock[F2,M1]	NU	1		1	-20
5	RespectStock[F2,M2]	B	1		2	
6	RespectStock[F2,M3]	NU	1		1	-11
7	RespectDemande[F1,D1]	NS	2	2	=	8
8	RespectDemande[F1,D2]	NS	1	1	=	3
9	RespectDemande[F2,D1]	B	0	-0	=	
10	RespectDemande[F2,D2]	NS	3	3	=	23
11	CoutTotal	B	52			

No.	Column name	St	Activity	Lower bound	Upper bound	Marginal
<hr/>						
1	D[F1,M1,D1]	NL	0	0		3
2	D[F1,M1,D2]	B	1	0		
3	D[F1,M2,D1]	B	1	0		
4	D[F1,M2,D2]	NL	0	0		24
5	D[F1,M3,D1]	B	1	0		
6	D[F1,M3,D2]	NL	0	0		10
7	D[F2,M1,D1]	NL	0	0		31
8	D[F2,M1,D2]	B	1	0		
9	D[F2,M2,D1]	NL	0	0		4
10	D[F2,M2,D2]	B	1	0		
11	D[F2,M3,D1]	NL	0	0		18
12	D[F2,M3,D2]	B	1	0		

Karush-Kuhn-Tucker optimality conditions:

```

49 KKT.PE: max.abs.err = 0.00e+00 on row 0
50         max.rel.err = 0.00e+00 on row 0
51         High quality

```

```

53 KKT.PB: max.abs.err = 0.00e+00 on row 0
54         max.rel.err = 0.00e+00 on row 0
55         High quality

```

```

57 KKT.DE: max.abs.err = 0.00e+00 on column 0
58         max.rel.err = 0.00e+00 on column 0
59         High quality

```

```

61 KKT.DB: max.abs.err = 0.00e+00 on row 0
62         max.rel.err = 0.00e+00 on row 0
63         High quality

```

End of output

1.2.3 Cas particulier 2

Données

$D \in \mathcal{M}_{n,n}(\mathbb{R})$ Matrice des distances

nClients le nombre de clients dans le magasin

Variables

On utilise une matrice $M \in \mathcal{M}_{n,n}(\{0,1\})$ avec n le nombre de clients et telle que

$$\forall 1 \leq i \leq n, \forall 1 \leq j \leq n, M_{i,j} = \begin{cases} 1 & \text{si l'on va du client } i \text{ vers le client } j \\ 0 & \text{sinon} \end{cases}$$

et un vecteur $u \in \mathcal{M}_n(\mathbb{N}), \forall 1 \leq i \leq n, u_i$ est la position du client C_i dans l'ordre de visite

Fonction objectif

$$f : \begin{cases} \mathcal{M}_{n,n}(\{0,1\}) \rightarrow \mathbb{R} \\ M \end{cases} \mapsto \min \left(\sum_{i=1}^n \sum_{j=1}^n M_{i,j} D_{i,j} \right)$$

Contraintes

On ne va chez un client qu'une seule fois $\forall 1 \leq i \leq n, \sum_{j=1}^n M_{i,j} = 1$

On ne sort d'un client qu'une seule fois $\forall 1 \leq j \leq n, \sum_{i=1}^n M_{i,j} = 1$

On ne fait pas de détour entre les clients

$$\forall 1 \leq i \leq n, \forall 1 \leq j \leq n, u_j + (\text{nClients} - 1) \geq u_i + \text{nClients} \times M(i, j)$$

Solution

Pour $D = \begin{pmatrix} 0 & 1 & 1 & 10 & 12 & 12 \\ 1 & 0 & 1 & 8 & 10 & 10 \\ 1 & 1 & 0 & 8 & 11 & 10 \\ 10 & 8 & 8 & 0 & 1 & 1 \\ 12 & 10 & 11 & 1 & 0 & 1 \\ 12 & 11 & 10 & 1 & 1 & 0 \end{pmatrix}$, la distance optimale résolue est DistanceTotale = 22.

La matrice solution est

$$\begin{pmatrix} & \text{Alpha} & \text{C1} & \text{C2} & \text{C3} & \text{C4} & \text{C5} \\ \text{Alpha} & 0 & 1 & 0 & 0 & 0 & 0 \\ \text{C1} & 0 & 0 & 0 & 1 & 0 & 0 \\ \text{C2} & 1 & 0 & 0 & 0 & 0 & 0 \\ \text{C3} & 0 & 0 & 0 & 0 & 1 & 0 \\ \text{C4} & 0 & 0 & 0 & 0 & 0 & 1 \\ \text{C5} & 0 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

Le chemin reconstruit est donc Alpha \rightarrow C1 \rightarrow C3 \rightarrow C4 \rightarrow C5 \rightarrow C2 \rightarrow Alpha

```
1 Problem: commerce
2 Rows: 43
3 Columns: 42 (42 integer, 36 binary)
4 Non-zeros: 182
5 Status: INTEGER OPTIMAL
6 Objective: DistanceTotale = 22 (MINimum)
7
```

No.	Row name	Activity	Lower bound	Upper bound
1	TousClientsServisUneFois[Alpha]	1	1	=
2	TousClientsServisUneFois[C1]	1	1	=
3	TousClientsServisUneFois[C2]	1	1	=
4	TousClientsServisUneFois[C3]	1	1	=
5	TousClientsServisUneFois[C4]	1	1	=
6	TousClientsServisUneFois[C5]	1	1	=
7	TousClientsQuittesUneFois[Alpha]	1	1	=
8	TousClientsQuittesUneFois[C1]	1	1	=
9	TousClientsQuittesUneFois[C2]	1	1	=
10	TousClientsQuittesUneFois[C3]	1	1	=
11	TousClientsQuittesUneFois[C4]	1	1	=
12	TousClientsQuittesUneFois[C5]	1	1	=
13	PasDeDetour[Alpha,C1]	-4	-4	
14	PasDeDetour[Alpha,C2]	5	-4	
15	PasDeDetour[Alpha,C3]	2	-4	
16	PasDeDetour[Alpha,C4]	3	-4	
17	PasDeDetour[Alpha,C5]	4	-4	
18	PasDeDetour[C1,C1]	0	-4	
19	PasDeDetour[C1,C2]	4	-4	
20	PasDeDetour[C1,C3]	-4	-4	
21	PasDeDetour[C1,C4]	2	-4	
22	PasDeDetour[C1,C5]	3	-4	
23	PasDeDetour[C2,C1]	-4	-4	
24	PasDeDetour[C2,C2]	0	-4	
25	PasDeDetour[C2,C3]	-3	-4	
26	PasDeDetour[C2,C4]	-2	-4	
27	PasDeDetour[C2,C5]	-1	-4	
28	PasDeDetour[C3,C1]	-1	-4	
29	PasDeDetour[C3,C2]	3	-4	
30	PasDeDetour[C3,C3]	0	-4	

70	31	PasDeDetour	[C3,C4]		
71				-4	-4
72	32	PasDeDetour	[C3,C5]		
73				2	-4
74	33	PasDeDetour	[C4,C1]		
75				-2	-4
76	34	PasDeDetour	[C4,C2]		
77				2	-4
78	35	PasDeDetour	[C4,C3]		
79				-1	-4
80	36	PasDeDetour	[C4,C4]		
81				0	-4
82	37	PasDeDetour	[C4,C5]		
83				-4	-4
84	38	PasDeDetour	[C5,C1]		
85				-3	-4
86	39	PasDeDetour	[C5,C2]		
87				-4	-4
88	40	PasDeDetour	[C5,C3]		
89				-2	-4
90	41	PasDeDetour	[C5,C4]		
91				-1	-4
92	42	PasDeDetour	[C5,C5]		
93				0	-4
94	43	DistanceTotale			
95				22	

No.	Column	name	Activity	Lower bound	Upper bound
99	1	M[Alpha,Alpha]			
100		*	0	0	1
101	2	M[Alpha,C1]	*	1	1
102	3	M[Alpha,C2]	*	0	1
103	4	M[Alpha,C3]	*	0	1
104	5	M[Alpha,C4]	*	0	1
105	6	M[Alpha,C5]	*	0	1
106	7	M[C1,Alpha]	*	0	1
107	8	M[C1,C1]	*	0	1
108	9	M[C1,C2]	*	0	1
109	10	M[C1,C3]	*	1	1
110	11	M[C1,C4]	*	0	1
111	12	M[C1,C5]	*	0	1
112	13	M[C2,Alpha]	*	1	1
113	14	M[C2,C1]	*	0	1
114	15	M[C2,C2]	*	0	1
115	16	M[C2,C3]	*	0	1
116	17	M[C2,C4]	*	0	1
117	18	M[C2,C5]	*	0	1
118	19	M[C3,Alpha]	*	0	1
119	20	M[C3,C1]	*	0	1
120	21	M[C3,C2]	*	0	1
121	22	M[C3,C3]	*	0	1
122	23	M[C3,C4]	*	1	1
123	24	M[C3,C5]	*	0	1
124	25	M[C4,Alpha]	*	0	1
125	26	M[C4,C1]	*	0	1
126	27	M[C4,C2]	*	0	1
127	28	M[C4,C3]	*	0	1
128	29	M[C4,C4]	*	0	1
129	30	M[C4,C5]	*	1	1
130	31	M[C5,Alpha]	*	0	1
131	32	M[C5,C1]	*	0	1

```

132      33 M[C5,C2]      *          1          0          1
133      34 M[C5,C3]      *          0          0          1
134      35 M[C5,C4]      *          0          0          1
135      36 M[C5,C5]      *          0          0          1
136      37 u[C1]         *         -4
137      38 u[Alpha]      *         -5
138      39 u[C2]         *          0
139      40 u[C3]         *         -3
140      41 u[C4]         *         -2
141      42 u[C5]         *         -1

```

```

142
143 Integer feasibility conditions:

```

```

144
145 KKT.PE: max.abs.err = 0.00e+00 on row 0
146         max.rel.err = 0.00e+00 on row 0
147         High quality

```

```

148
149 KKT.PB: max.abs.err = 0.00e+00 on row 0
150         max.rel.err = 0.00e+00 on row 0
151         High quality

```

```

152
153 End of output

```