

Introductory Lecture

Consider that we have a set of *linear constraints* and an *objective function* that we want to *optimise*. Example One: Portfolio Analysis. An investor has £600,000 to allocate among the *alternatives* shown. The investor requires: (1) The **full** amount should be invested; (2) The **average** risk should be ≤ 3.5 ; (3) At least £100,000 in govt. bonds, £100,000 in money market, £60,000 in hi-tech and £50,000 in municipal bonds. How should the investor *maximise* the return of the investment?

	Rate of Return	Risk Factor
A. Govt. Bonds	7.5%	0
B. Money Market	8.6%	2
C. Hi-tech Stock	13.5%	9
D. Municipal Stock	7.8%	1

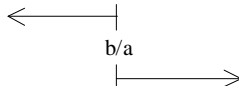
Let x_A be the amount *invested* in option A, etc. Constraints: $x_A \geq 100,000$; $x_B \geq 100,000$; $x_C \geq 60,000$; $x_D \geq 50,000$; $x_A + x_B + x_C + x_D = 600,000$; and $(x_A/600000).0 + (x_B/600000).2 + (x_C/600000).9 + (x_D/600000).1 \leq 3.5$, **or** $2x_B + 9x_C + x_D \leq (3.5)600000$. The objective function is the *annual return*, $1/100(7.5x_A + 8.6x_B + 13.5x_C + 7.8x_D)$.

	Dist. 1	Dist. 2	Capacity
Plant 1	£5 (x_1)	£8 (x_2)	≤ 500
Plant 2	£10 (x_3)	£9 (x_4)	≤ 600
Demands	≥ 400	≥ 650	

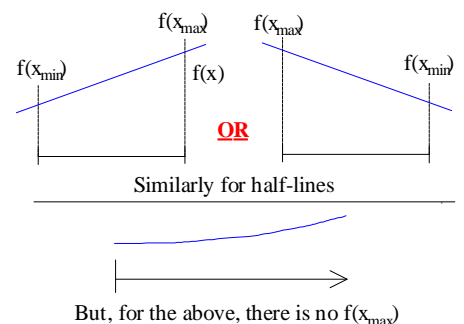
Example Two: Transportation. A company makes water heaters at 2 plants, and distributes from two centres as shown. (The £x in the cells denote the **cost** of transportation). The company wants to *minimise* its transportation costs. Constraints: $x_1 + x_2 \leq 500$, $x_3 + x_4 \leq 600$, $x_1 + x_3 \geq 400$, and $x_2 + x_4 \geq 650$. **Cost Function** (to be minimised): $C(x_1, x_2, x_3, x_4) = 5x_1 + 8x_2 + 10x_3 + 9x_4$.

Characteristics of L.P. Problems

The constraints are linear. If x_1, \dots, x_n are the *variables*, then all constraints have the form $a_1x_1 + a_2x_2 + \dots + a_nx_n \leq b$ (or $\geq b$, or $= b$) for some *real numbers* a and b. (We can use the **shorthand** $a.x \leq b$). **The objective function** is linear: $f(x_1, \dots, x_n) = c_1x_1 + c_2x_2 + \dots + c_nx_n$. The problem is to find some *point* $x_{OPT} \in \mathbf{R}^n$ s.t. the *constraints* are satisfied and $f(x_{OPT})$ is “*optimal*”.

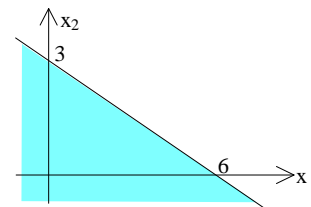
Linear constraints in low dimensions. *Idiot example:* $n = 1$. Constrain $a.x \leq b$ (or $a.x \geq b$). If $a \neq 0$, and say $a > 0$, then $x \leq b/a$. If $a < 0$, then $x \geq b/a$, i.e.  we get a *half-line* as shown. If $x \leq b_1$ and $x \leq b_2$, we get *redundancy*: $x \leq \min(b_1, b_2)$, $b_2 \leq b_1$ etc. If $x \leq b_1$ and $x \geq b_2$, then we get the *interval* as shown on the left, provided that $b_2 \leq b_1$ (or **empty** if $b_1 < b_2$). The set of *feasible points* may be empty.

What about linear functions? The *feasible region* is an **interval**, a **half line**, or **empty**. A linear function has the *form* $f(x) = cx (+ d)$ [Note: the +d is not going to change where x_{OPT} is]. **Notes:** There may be *no* feasible points; some constraints may be *redundant*; *no optima* need exist if the region is unbounded; and *more than one* optimal point may exist (when $f(x)$ is a **constant** (straight line)).

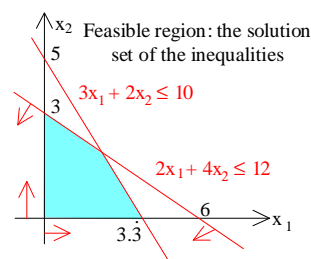


n = 2. A linear constraint is now of the form $a_1x_1 + a_2x_2 \leq b$. This determines a half plane on one or the other side of the line $a_1x_1 + a_2x_2 = b$.

Example: $2x_1 + 4x_2 \leq 12$.



Typical Example: A paint manufacturer produces 2 types of paint — standard quality (S) and top quality (T). Two ingredients are used, pigment and resin. S requires 2 units of pigment and 3 units of resin for each unit sold at a profit of £1 a unit. T requires 4 units of pigment and 2 units of resin for each unit sold at a profit of £1.50 a unit. The stocks are 12 units of pigment and 10 units of resin. How much of each type of paint should be made to maximise the profit?

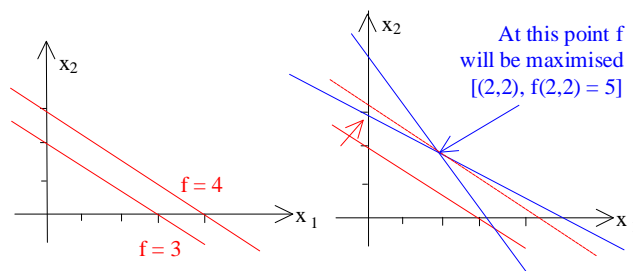


3rd February 2000

Variables: Let x_1 denote the number of units of type S; x_2 of type T. **Pigment constraint:** $2x_1 + 4x_2 \leq 12$; **Resin constraint:** $3x_1 + 2x_2 \leq 10$. Also, $x_1 \geq 0$ and $x_2 \geq 0$. **Objective function:** $1x_1 + 1.5x_2$. **Note:** the feasible region is convex — not as on the right. Each constraint gives a convex “half space”. The **intersection** of convex sets is convex.



For any given real number, c , there are **infinitely** many points (x_1, x_2) where $x_1 + \frac{3}{2}x_2 = c$. The lines with **constant** value are parallel to each other. To increase the value, we imagine moving the lines (*North East!*) as far as possible, i.e. as long as the line *intersects* the feasible region. It seems that **max** (and **min**) values occur at vertices, but could also occur all along a **single** “edge” when $f(\underline{x}) = \text{constant}$ is *parallel* to some edge, *face*, etc.



Description of the Feasible Region

The feasible region is *always convex*. **Definition:** $S \subseteq \mathbf{R}^n$ is convex if given $\underline{x}, \underline{y} \in S$, and $t \in [0,1]$, then $t\underline{x} + (1-t)\underline{y} \in S$. (The line joining any two points in S lies wholly **within** S). **Lemma:** If $\underline{l} \cdot \underline{x} \leq c$ is a linear constraint, then $S = \{\underline{x} \mid \underline{l} \cdot \underline{x} \leq c\}$ is **convex**. **Proof:** If $\underline{x}, \underline{y} \in S$, then $\underline{l} \cdot \underline{x} \leq c$, and $\underline{l} \cdot \underline{y} \leq c$. Let $0 \leq t \leq 1$, and let $\underline{x}_t = t\underline{x} + (1-t)\underline{y}$. Check if $\underline{l} \cdot \underline{x}_t \leq c$: $\underline{l} \cdot \underline{x}_t = t\underline{l} \cdot \underline{x} + (1-t)\underline{l} \cdot \underline{y} \leq tc + (1-t)c = c$.

Lemma: If S_1, \dots, S_k are convex subsets of \mathbf{R}^n , then so is $\bigcap_{i=1}^k S_i$. This implies that all *feasible regions* are convex. **Fact:** A linear function defined on a *convex set* takes its *maximum* and *minimum* values at **vertices**.

Summary of the situation (in general): the feasible region is *convex*, but may be **unbounded** or may be **empty**. The optimal value will occur at a *vertex*, but (i) one may not exist (i.e. on **unbounded** regions); (ii) the solution may not be *unique* (if M is the max value, and $f(\underline{x}) = M$ and $f(\underline{y}) = M$, then **all** points on the line joining \underline{x} and \underline{y} *also* take this value).

Algorithmic Approach

1st Attempt (Not practical): (i) find the **vertices**, v ; (ii) **evaluate** $f(v)$ for each v ; (iii) find the **biggest!** **New idea**: find 1 feasible vertex, v_1 , and then test to see which direction is *best* to increase f the most. Find the (*first*) feasible vertex in that direction; and go to that vertex and **repeat**. When you can't increase f any further, *stop*. To aid interpretation, we introduce slack & surplus variables, and write all inequalities in the **same** form.

Example: $2x_1+4x_2 \leq 12$. Introduce s_1 , a slack variable, so that $2x_1+4x_2+s_1 = 12$, with $s_1 \geq 0$. **Paint Example**: $2x_1+4x_2+s_1 = 12$, and $3x_1+2x_2+s_2 = 10$, with $x_1 \geq 0$, $x_2 \geq 0$, $s_1 \geq 0$ and $s_2 \geq 0$. We now have the “*equations*” in the form $A\underline{x} = \underline{b}$. (*m equations in n variables*). A basic solution to $A\underline{x} = \underline{b}$ is obtained by setting $(n-m)$ variables equal to 0, and **solving** for the remaining m (*basic*) variables. At a vertex, the $(n-m)$ variables that are 0 are called non-basic variables. Those that are non-zero are called basic.

Simplex method (*long-winded* version). *Maximise* $x_1+1.5x_2+0s_1+0s_2 = M$ (**OBJ**) subject to $2x_1+4x_2+s_1 = 12$ (**C1**) and $3x_1+2x_2+s_2 = 10$ (**C2**). **All** variables are ≥ 0 . **Step 1**: Find a “*start*” vertex. Here, it's clear that $x_1 = x_2 = 0$ is a *feasible* solution, so we can take it to be the start vertex. x_1, x_2 *non-basic*; s_1, s_2 *basic*.

Step 2: **Rewrite** M and the basic variables in terms of the *non-basic* ones to get **one** basic variable in each, with coefficient 1: $M-x_1-1.5x_2 = 0$ (**OBJ A**); $2x_1+4x_2+s_1 = 12$ (**C1 A**); and $3x_1+2x_2+s_2 = 10$ (**C2 A**). **Step 3**: Can we *increase* M ? If so, how? (Increasing x_2 increases M **most quickly**, as the coefficient of x_2 is the *most -ve* in (**OBJ A**)).

Step 4: Increase the chosen variable *as much as possible*, keeping the other non-basic variables at **zero**, and satisfying (**C1 A**) and (**C2 A**). (**C1 A**) gives $s_1 = 12-2x_1-4x_2$. If $x_1 = 0$ and $s_1 \geq 0$, then $x_2 \leq 12/4 = 3$. (**C2 A**) similarly gives $x_2 \leq 10/2 = 5$ (Ratio: $\frac{\text{constant}}{\text{coefficient}}$). We can **only** increase x_2 by 3. (x_1, s_1 *non-basic*; x_2, s_2 *basic*). Now return to step 2 and **repeat**.

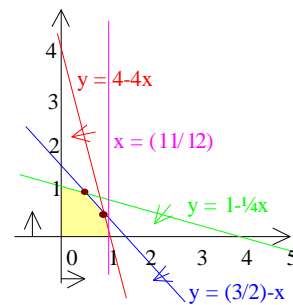
Step 2 (2nd pass): (**C1 A**)/4 gives $\frac{1}{2}x_1+x_2+\frac{1}{4}s_1 = 3$ (**C1 B**). (**C2 A**) - 2(**C1 B**) gives $2x_1-\frac{1}{2}s_1+s_2 = 4$ (**C2 B**). And $M-\frac{1}{4}x_1+\frac{3}{8}s_1 = 4.5$ (**OBJ B**) [x_1 and s_1 are zero, **so** $M = 4.5$ here]. **Step 3 (Again)**: Increase x_1 to *increase* M , keeping $s_1 = 0$. (**C1 B**) gives $x_1 \leq 3/\frac{1}{2} = 6$. (**C2 B**) gives $x_1 \leq 4/\frac{1}{2} = 2$. (This is *stronger*). We can **only** increase x_1 to 2. So $x_1 = 2$, $s_1 = 0$, $s_2 = 2$ and $s_2 = 0$.

Now repeat 2, 3 and 4. (**C2 B**) $\rightarrow x_1-\frac{1}{4}s_1 = 2$ (**C2 C**). (**C1 B**) - $\frac{1}{2}$ (**C2 C**) gives $x_2+\frac{3}{8}s_1-\frac{1}{4}s_2 = 2$ (**C1 C**). And $M+\frac{5}{16}s_1+\frac{1}{8}s_2 = 5$. Because there are **no -ve coefficients**, we have reached a maximum: **$M_{\text{MAX}} = 5$** . Here, $s_1 = s_2 = 0$, and **$x_1 = 2, x_2 = 2$** .

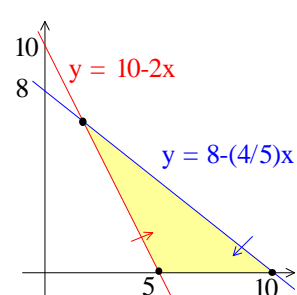
Tutorial

Q: By **graphical** means, find the *maximum* value of the function $f = 2x+y$ subject to $2x+2y \leq 3$, $4x+y \leq 4$, and $x+4y \leq 4$. ($x \geq 0, y \geq 0$). Add the *constraint* $12x \leq 11$ and redo.

A: First change the *inequalities* to $y = \dots$ ones, i.e. $y \leq 3/2 - x$, $y \leq 4 - 4x$, and $y \leq 1 - 1/4x$. From the *diagram* shown on the right, we have **2** vertices to consider. **Vertex 1**: find the co-ordinates by solving: $4 - 4x = 3/2 - x$; $5/2 = 3x$; $x = 5/6$, so $y = 2/3$. Here, $f = 7/3$. At the **other vertex**, (solving) $3/2 - x = 1 - 1/4x$, so $x = 4/6$; $y = 5/6$; and $f = 13/6$ (*less*). So the **minimum** occurs at $(5/6, 2/3)$. Adding the (**purple**) *constraint* does not **change** the solution, as no vertices are “cut off”.



Q: A **food** mixture for chickens is made using foods A and B. Each unit of A weighs 4 grams, and contains 2 grams of *protein*. Each unit of B weighs 5 grams, and contains 1 gram of *protein*. The mixture weighs **at most** 40 grams, and must contain **at least** 10 grams of protein. Assume that A costs 30p a unit, and B 10p a unit. By graphical means, find the values that **minimise** the cost of the mixture used.



A: Here, $4x + 5y \leq 40$ ($y \leq 8 - 4/5x$) and $2x + y \geq 10$ ($y \geq 10 - 2x$) are the *constraints*. The objective function (OBJ) is $30x + 10y = M$. **Assume** that $x \geq 0$ and $y \geq 0$. Looking at the *graph*, vertex $(5, 0)$ gives $M = 150$. Vertex $(10, 0)$ gives $M = 300$. The **other** vertex occurs at (solving) $10 - 2x = 8 - 4/5x$; $x = 5/3$, so $y = 20/3$. Here, $M = 350/3$ (*smaller*), so that $(5/3, 20/3)$ is the *minimum* point.

10th February 2000

Simplex Method (Tableau)

M	x_1	x_2	s_1	s_2		
C ₁	0	2	4	1	0	12
C ₂	0	3	2	0	1	10
OBJ	1	-1	-3/2	0	0	0

ratios
 $12/4 = 3$ ← smallest ratio, pivot row
 $10/2 = 5$

pivot element (4)
 pivot (largest column ve value)

C _{1B}	0	1/2	1	1/4	0	3
C _{2B}	0	2	0	-1/2	1	4
OBJ	1	-1/4	0	3/8	0	4.5

ratios
 $3/2 = 6$
 $4/2 = 2$ ← pivot row

new pivot column (Not optimal because of the -1/4)

C _{1C}	0	0	1	3/8	-1/4	2
C _{2C}	0	0	1	0	-1/4	2
OBJ	1	0	0	5/16	1/8	5

This method, shown on the left, codes up the previous calculations. After the first table, divide the pivot row by the **pivot** element, and use it to *eliminate* other entries in the pivot column. After the **3rd** table, there are no negative entries in the (OBJ) row, so we have an optimal tableau: $M_{\text{MAX}} = 5$, $x_1^* = 2$, and $x_2^* = 2$. There is no *slack*, because $s_1^* = s_2^* = 0$. (A “*” indicates the value of the variable at the *optimum* position.)

Variants. (a) Minimisation. Can be done by (i) minimising [-objective function] (*minus*). **Beware:** this often requires “*pre-processing*” to find an initial vertex to start from. (ii) Use a dual program which is a maximisation — see **later!**

(b) Finding an initial vertex. Here is an **ad hoc** method, which works moderately well with small examples, but has no guarantee of success. **Example** (illustrating both): 3 mines produce 3 grades of coal as shown in the *table*. (The numbers shown show the production *per day* in tons).

Quality	Mine 1	Mine 2	Mine 3
High	2	6	4
Medium	3	2	6
Low	8	3	2

The operating costs *per day* are £100 for Mine 1, £120 for Mine 2, and £140 for Mine 3. There is an order for 20 tons of high grade coal, 20 tons of medium, and 30 tons of low. Find a way to **meet** the order, **minimising** the costs.

Let Mine 1 be operated for x_1 days, etc. So we have $2x_1+6x_2+4x_3 \geq 20$; $3x_1+2x_2+6x_3 \geq 20$; $8x_1+3x_2+2x_3 \geq 30$; and cost $N = 100x_1+120x_2+140x_3$. To *minimise* N , we **maximise** $M = (-N) = -100x_1-120x_2-140x_3$. *Slack* variables: $2x_1+6x_2+4x_3-s_1 = 20$; $3x_1+2x_2+6x_3-s_2 = 20$; and $8x_1+3x_2+2x_3-s_3 = 30$ (with $x_1, x_2, x_3, s_1, s_2, s_3 \geq 0$).

In the **first** table, we set out the problem, remembering that $x_1 = x_2 = x_3 = 0$, and that s_1, s_2 and s_3 are < 0 . But this then means that we are not at a **feasible** position. To *search* for a feasible position, try to eliminate one of the x_i 's (as in *Gaussian Elimination*).

After we do this, and reach the *second* table, $x_2 = x_2 = s_1 = 0$, and $x_1 = s_1 = 10$, and $s_3 = 50$ (i.e. *feasible*). So we **carry** on. (Note: **blue** means pivot row/column, **red** means pivot element). After we finish, we have the **optimal** solution: Run Mine 1 for $2^{6/7}$ days; Mine 2 for $1^{3/7}$ days; and Mine 3 for $1^{3/7}$ days. (Note: **no** surplus). The *minimal* cost is $\pounds 657^{1/7}$ (**Max** value of $M = -$ **Minimal** value of N).

	M	x_1	x_2	x_3	s_1	s_2	s_3	ratios
C ₁	0	2	6	4	-1	0	0	20
C ₂	0	3	2	6	0	-1	0	20
C ₃	0	8	3	2	0	0	-1	30
OBJ	1	100	120	140	0	0	0	0
C _{1/2}	0	1	3	2	$-\frac{1}{2}$	0	0	10 $\frac{10}{3}$
	0	0	-7	0	$\frac{3}{2}$	-1	0	-10 $\frac{10}{7}$
	0	0	-21	-14	4	0	-1	-50 $\frac{50}{21}$
	1	0	-180	-60	50	0	0	-1000
	0	1	0	2	$\frac{1}{7}$	$-\frac{3}{7}$	0	$\frac{40}{7}$ $\frac{40}{14}$
	0	0	1	0	$-\frac{3}{14}$	$\frac{1}{7}$	0	$\frac{10}{7}$ \times
	0	0	0	-14	$-\frac{1}{2}$	3	-1	-20 $-\frac{20}{14}$
	1	0	0	-60	$\frac{80}{7}$	$\frac{180}{7}$	0	$-\frac{5200}{7}$
	0	1	0	0	$\frac{1}{14}$	0	$-\frac{1}{7}$	$\frac{20}{7}$
	0	0	1	0	$-\frac{3}{14}$	$\frac{1}{7}$	0	$\frac{10}{7}$
	0	0	0	1	$\frac{1}{28}$	$-\frac{3}{14}$	$\frac{1}{4}$	$\frac{10}{7}$
	1	0	0	0	$\frac{95}{7}$	$\frac{90}{7}$	$\frac{30}{7}$	$-\frac{4600}{7}$

Assignment 1: Set 9/2; In 18/2; Back 23/2

Q: **Maximise** $M = 7x_1+9x_2+3x_3$, *subject to* $x_1+4x_2+8x_3 \leq 40$, $2x_1+x_2+6x_3 \leq 60$, and $x_1+x_2+x_3 \leq 16$, with $x_1 \geq 0$, $x_2 \geq 0$, and $x_3 \geq 0$. A: Introducing *slack variables* s_1, s_2 and s_3 to eliminate the need for inequalities, we obtain the following set of **equations**: $x_1+4x_2+8x_3+s_1 = 40$, $2x_1+x_2+6x_3+s_2 = 30$, $x_1+x_2+x_3+s_3 = 16$, and $M = 7x_1+9x_2+3x_3$.

Solving the question using *tableau*, we start off with the first tableau as shown. Because -9 is the largest negative entry in the bottom row, the values in that column constitute our first pivot **column**. Tip: in the exam, say this on the sheet. Calculating the ratios, because 10 is the smallest **positive** ratio, then we take this to be the *pivot row*. Say this also. So 4 is the pivot element shown, and we *divide the pivot row by 4* to obtain the second tableau.

Next, we perform *row2 - row1*, *row3 - row1*, and *row4 + 9×row1* to get the **third** tableau. There are *negative* entries in the bottom row, so the tableau is **not** optimal, and so we need to find a new pivot element as shown ($\frac{3}{4}$). *Dividing* the pivot row by $\frac{3}{4}$ obtains the fourth tableau, and doing *row1 - $\frac{1}{4}$ row3*, *row2 - $\frac{7}{4}$ row3*, and *row4 + $\frac{19}{4}$ row3*, we obtain the *final* tableau, which is optimal because there are **no** negative entries in the final row. So the *optimal* values are $x_1^* = 8$, $x_2^* = 8$, $x_3^* = 0$, $s_1^* = 0$, $s_2^* = 6$, $s_3^* = 0$, and $M_{\text{MAX}} = 128$.

	M	x_1	x_2	x_3	s_1	s_2	s_3	ratios
C ₁	0	1	4	8	1	0	0	40 10
C ₂	0	2	1	6	0	1	0	30 30
C ₃	0	1	1	1	0	0	1	16 16
OBJ	1	-7	-9	-3	0	0	0	0
C _{1/4}	0	$\frac{1}{4}$	1	2	$\frac{1}{4}$	0	0	10
	0	2	1	6	0	1	0	30
	0	1	1	1	0	0	1	16
	1	-7	-9	-3	0	0	0	0
	0	$\frac{1}{4}$	1	2	$\frac{1}{4}$	0	0	10 $\frac{40}{7}$
	0	$\frac{7}{4}$	0	4	$-\frac{1}{4}$	1	0	20 $\frac{80}{7}$
	0	$\frac{3}{4}$	0	-1	$-\frac{1}{4}$	0	1	6 $\frac{6}{3/4}$
	1	$-\frac{19}{4}$	0	15	$\frac{9}{4}$	0	0	90
	0	$\frac{1}{4}$	1	2	$\frac{1}{4}$	0	0	10
	0	$\frac{7}{4}$	0	4	$-\frac{1}{4}$	1	0	20
C _{3^{3/4}}	0	1	0	$-\frac{4}{3}$	$-\frac{1}{3}$	0	$\frac{4}{3}$	8
	1	$-\frac{19}{4}$	0	15	$\frac{9}{4}$	0	0	90
	0	0	1	$\frac{7}{3}$	$\frac{1}{3}$	0	$-\frac{1}{3}$	8
	0	0	0	$\frac{19}{3}$	$\frac{1}{3}$	1	$-\frac{7}{3}$	6
	0	1	0	$-\frac{4}{3}$	$-\frac{1}{3}$	0	$\frac{4}{3}$	8
	1	0	0	$\frac{26}{3}$	$\frac{2}{3}$	0	$\frac{19}{3}$	128

Duality

Every L.P. problem has a **dual** problem. **Example** (Paint): maximise $x_1 + 1.5x_2$ subject to $2x_1 + 4x_2 \leq 12$ and $3x_1 + 2x_2 \leq 10$ ($x_1, x_2 \geq 0$). **Check**: if it is a *maximisation* problem, all nontrivial constraints take the form $\underline{a} \cdot \underline{x} \leq c$ (*minimisation*: $\underline{a} \cdot \underline{x} \geq c$). The number of **variables** in the dual problem is equal to the number of nontrivial **constraints** in the “*primal*” (i.e. *original*) problem.

Associate u_1 with $2x_1 + 4x_2 \leq 12$; and u_2 with $3x_1 + 2x_2 \leq 10$. **Primal** max \Leftrightarrow **Dual** min; **Primal** min \Leftrightarrow **Dual** max. The *coefficients* of u_1, u_2 , etc. in the *dual objective function* are the corresponding right hand sides of the *constraints*, e.g. $12u_1 + 10u_2$. The number of nontrivial **constraints** in the dual is equal to the number of **variables** in the primal. $x_1 \rightarrow 2u_1 + 3u_2 \geq 1$; $x_2 \rightarrow 4u_1 + 2u_2 \geq 1.5$.

The dual of the **paint** problem is to *minimise* $12u_1 + 10u_2$ subject to $2u_1 + 3u_2 \geq 1$ and $4u_1 + 2u_2 \geq 1.5$, with u_1 and $u_2 \geq 0$. The **dual** of the *dual* is the primal! If a constraint is an **equality** constraint, e.g. $2x_1 + 4x_2 = 12$, *convert to* $2x_1 + 4x_2 \leq 12$ and $-2x_1 - 4x_2 \leq -12$.

Solving the problem using the *tableaux* shown, we see that the **first** tableau is not feasible. To *correct* this, we perform $(R_2 - 2R_1)$, $(OBJ - 6R_1)$, and $(\frac{1}{2}R_1)$, to get the *second* tableau, which *is* feasible. We then find the **pivot** element (shown in **red**), and then find the *final* tableau, which is the **optimal** tableau. Thus at $u_1 = \frac{5}{16}$, $u_2 = \frac{1}{3}$, and $s_1 = s_2 = 0$, we have $-OBJ = 5$. *Comparing* solutions, (**Primal**) $\{x_1 = 2, x_2 = 2, s_1 = 0, s_2 = 0, OBJ_{MAX} = 5\}$; *coefficients* of s_1 and s_2 in OBJ_{MAX} : $\frac{5}{16}, \frac{1}{8}$; (**Dual**) $\{u_1 = \frac{5}{16}, u_2 = \frac{1}{3}, s_1 = 0, s_2 = 0, OBJ_{MIN} = 5\}$; *coefficients* of s_1 and s_2 in OBJ_{MIN} : $2, 2$.

	M	u_1	u_2	s_1	s_2		ratios
C_1	0	2	3	-1	0		1
C_2	0	4	2	0	-1		1.5
OBJ	1	12	10	0	0		0
	0	1	1.5	-0.5	0		0.5
	0	0	-4	2	-1		-0.5
	1	0	-8	6	0		-6
	0	1	0	0.25	-0.375		$\frac{5}{16}$
	0	0	1	-0.5	0.25		$\frac{1}{3}$
	1	0	0	2	2		-5

Duality Theorem

(1) If either the *primal* problem or the *dual* problem has an optimal solution, then they both have **optimal** solutions. If an optimal solution exists, the primal and dual objective functions have the same optimal value. (2) The **optimal** solution values of the primal variables (x_1, x_2 , etc.) are given by the *coefficients* corresponding to the slack variables in the **dual** optimal tableau.

Principle of Complementary Slackness

At the *solution* point to the primal and dual problems, either a variable has **zero** value, or the *constraint* associated with that variable (in the dual problem) is an **equality** constraint (when the optimum solution values are substituted in). (**Paint**) The *optimal* solution values x_1 and x_2 are both **non-zero**, and the “surplus/slack” variables in the *dual* **are** zero, i.e. the “*inequalities are equalities*”. The *optimal* values of u_1 and u_2 are non-zero, while the corresponding **inequalities** are **equalities**.

Interpretation of duality: in the *paint* example, both the pigment and the resin have a value given indirectly by the *profit* obtained by selling the product. How can this be **evaluated**?

Shadow Prices

The *optimal values* of the dual variables are called shadow prices. A shadow price represents the amount of change in the optimal value of an *objective function* resulting from a **unit** change in the right hand side value of a *primal* constant. **Caution**: changing constraints may change the *feasibility* of the optimal solution. A shadow price is therefore only valid for a **specific** interval within which a right hand side value can *change* without this happening.

Example: minimise $Z = 4x_1 + 9x_2$ subject to $x_1 + 2x_2 \geq 200$ and $2x_1 + 6x_2 \geq 750$, with x_1 and $x_2 \geq 0$. Looking at the diagram, the *dual problem* is $y_1 + 2y_2 \leq 4$ (*constraint 1*) and $2y_1 + 6y_2 \leq 9$ (*constraint 2*). We need to **maximise** $200y_1 + 750y_2$.

$$\begin{array}{cc|cc|c} 1 & 2 & 200 & 1 & 2 & 4 \\ 2 & 6 & 750 & 2 & 6 & 9 \\ \hline 4 & 9 & & 200 & 750 & \end{array}$$

Using the *simplex* method, we get the **optimal** tableau as shown on the left. Here, $y_1^* = 0$, $y_2^* = 3/2$, $s_1^* = 1$, and $s_2^* = 0$. The solution to the *primal* problem is $x_1^* = 0$, $x_2^* = 125$, and $OBJ^* = 1125$. Back in the *primal* problem, **Constraint 1** gives $x_1^* + 2x_2^* = 250 \geq 200$. The surplus of 50 is the *coefficient of* y_1^* in the dual OBJ. **Constraint 2** gives $2x_1^* + 6x_2^* = 750$: no surplus (= the *coefficient of* y_2^* in dual OBJ).

Notice that the **dual** objective function is $200y_1 + 750y_2$, and that the *optimal values* of y_1 and y_2 are 0 and $3/2$. Therefore, $OBJ^{OPT} = 200(0) + 750(3/2) = 1125$. If we vary the **RHS** of the original constraints, the coefficients of the *corresponding* y_i in the dual objective function vary. For example, **varying** the 200 to 201 does not *change* the minimum cost — but varying 750 to 751 changes OBJ^{OPT} by £1.50. (This is the *corresponding* shadow value).

21st February 2000

Mixed Constraints / Big M Method

Example: maximise $Z = 3x_1 + 4x_2 + x_3$ subject to $x_1 + 2x_2 + 3x_3 \leq 8$ and $2x_1 + x_2 + 4x_3 \geq 10$. To see *what happens*, carry on regardless: add **slack** and **surplus** variables, giving $x_1 + 2x_2 + 3x_3 + s_1 = 8$ and $2x_1 + x_2 + 4x_3 - s_2 = 10$. **But** $x_1 = x_2 = x_3 = 0$ is *not a feasible solution*: $s_1 = 8$, but $s_2 = -10$ (oh no!). So introduce an **artificial** variable a_1 to absorb the -ve value, but to ensure that it cannot *influence* the optimal value, you add $-Ma_1$ to Z , where M is an **arbitrarily large** +ve constant. So our *new* problem is to maximise $Z = 3x_1 + 4x_2 + x_3 - Ma_1$ subject to $x_1 + 2x_2 + 3x_3 + s_1 = 8$ and $2x_1 + x_2 + 4x_3 + a_1 - s_2 = 10$, with $x_1, x_2, x_3, s_1, s_2, a_1 \geq 0$. Now **solve**.

Sensitivity Analysis

After an *optimal* solution to an L.P. problem has been found, it is useful to study the effects of **small changes** to the original problem, for example (1) changes to the *objective function coefficients*; (2) changes to the *RHS of constraints*; (3) changes in the *constraint coefficients*; (4) the inclusion of *additional variables* to the problem; and (5) the inclusion of *additional constraints*. We will only look at (1) and (2).

Tutorial

Q: **Minimise** $Z = 600x_1 + 460x_2 + 600x_3 + 800x_4$ subject to $2x_1 + x_2 + x_3 \geq 40$, $x_1 + 2x_2 + x_3 + x_4 \geq 60$, and $2x_1 + 2x_2 + x_4 \geq 50$, where $x_1, x_2, x_3, x_4 \geq 0$. A: First find the *dual* problem: it is to **maximise** $40u_1 + 60u_2 + 50u_3 = M$ s.t. $2u_1 + u_2 + 2u_3 \leq 600$, $u_1 + 2u_2 + 2u_3 \leq 460$, $u_1 + u_2 \leq 600$, and $u_2 + u_3 \leq 800$.

We can solve this using the *normal* method, and when we finish, we have the optimal tableau as shown on the **right**. Now the *optimal solution* of the dual is as follows: $u_1^* = 740/3$, $u_2^* = 320/3$, $u_3^* = 0$, $s_1^* = s_2^* = 0$, $s_3^* = 740/3$, $s_4^* = 2080/3$, and $Z_{OPT} = 48800/3$.

0	1/2	0	1/3	1/3	-1/6	0	0	370/3
0	0	1	2/3	-1/3	2/3	0	0	320/3
0	0	0	-4/3	-1/3	-1/3	1	0	740/3
0	0	0	1/3	1/3	1/3	0	1	2080/3
1	0	0	50/3	20/3	80/3	0	0	48800/3

The **primal** solution (look at the *coefficients*) is $x_1^* = 20/3$, $x_2^* = 80/3$, and $x_3^* = x_4^* = 0$. Now look at the amount of *surplus* in the **primal** constraints. Firstly, u_1 : $2(20/3) + 80/3 + 0 = 40$, so *no surplus* (look at the **red** equation to see where *that* came from). Secondly, u_2 : $20/3 + 2(80/3) + 0 + 0 = 60$, so *no surplus*. Finally, u_3 : $2(20/3) + 2(80/3) + 0 = 200/3$, so a *surplus* of $200/3 - 50 = 50/3$.

Identify and **interpret** the *shadow prices*: these represent the *increase* in M_{OPT} resulting from a change in the RHS of a **primal** constraint. Changing 40 to 41 in $40u_1 + 60u_2 + 50u_3 = M$ gives a *shadow price* of $u_1^* = 740/3$, because $M_{OPT} = 48800/3 = 40(740/3) + 60(320/3) + 50(0)$, so the *shadow price* is $48800/3 - 41(740/3) - 60(320/3) - 50(0) = 740/3$. *Similarly*, the shadow price of u_2^* is $320/3$, while the shadow price of u_3^* is 0.

Assignment 2: Set 23/2; In 3/3; Back 10/3

Q: **Minimise** $Z = 110x_1 + 120x_2 + 150x_3$ subject to $x_1 + x_2 \geq 4$, $x_1 + x_3 \geq 6$, $x_1 + x_2 + x_3 \geq 5$, and $x_3 \geq 1$, where $x_1, x_2, x_3 \geq 0$, using the *dual simplex algorithm*. (**Find** the dual problem; **solve** the dual problem; find the **solution** to the primal problem from the **dual** tableau; state the amount of **surplus** in each primal constraint at the optimal point; and identify and interpret the **shadow prices**).

M	u ₁	u ₂	u ₃	u ₄	s ₁	s ₂	s ₃	ratios
0	1	0	1	0	1	0	0	110 110
0	1	0	1	0	0	1	0	120 ×
0	0	1	1	1	0	0	1	150 150
1	-4	-6	-5	-1	0	0	0	0

A: The **dual** problem is to *maximise* $z = 4u_1 + 6u_2 + 5u_3 + u_4$ subject to $u_1 + u_2 + u_3 \leq 110$, $u_1 + u_3 \leq 120$, and $u_2 + u_3 + u_4 \leq 150$, with $u_1, u_2, u_3, u_4 \geq 0$. Solving the dual problem, we start with the *first* tableau on the right. Here, -6 is the pivot **column** (the largest -ve entry), and 110 is the pivot **row** (the smallest +ve ratio). So doing row3-row 1 and row 4+6row 1 to produce *zeroes* in the pivot column, we obtain the second tableau.

0	1	1	1	0	1	0	0	110 ×
0	1	0	1	0	0	1	0	120 ×
0	-1	0	0	0	-1	0	1	40 40
1	2	0	1	-1	6	0	0	660
0	1	1	1	0	1	0	0	110
0	1	0	1	0	0	1	0	120
0	-1	0	0	1	-1	0	1	40
1	1	0	1	0	5	0	1	700

Note that the tableau is **not** optimal because the last row contained a *negative* entry. In the second tableau, -1 is the pivot **column** (the largest -ve entry), and 40 is the pivot **row** (the smallest +ve ratio). So we do row 4+row 3 to produce *zeroes* in the pivot column, and in doing so obtain the third tableau.

Because the objective row doesn't contain a **negative** entry, the tableau is optimal. Therefore, the solution of the *dual* problem is as follows: $u_1^* = 0$, $u_2^* = 110$, $u_3^* = 0$, $u_4^* = 40$, $s_1^* = 0$, $s_2^* = 120$, $s_3^* = 0$, and $z_{OPT} = 700$. The solution to the **primal** problem can be obtained *from the dual tableau* as follows:

The coefficients of the slack variables in the *optimal* tableau (in the objective row) give the optimal values of the primal variables. **Therefore**, $x_1^* = 5$, $x_2^* = 0$, and $x_3^* = 1$. The optimal value is *unchanged*, so that $Z_{OPT} = 700$. We can obtain the amount of *surplus* in each primal constraint at the optimal point by substituting the optimal **values** for the x_i into the primal equations (right at the beginning of the question).

For example, for the **first** constraint, $x_1 + x_2 \geq 4$, as $x_1^* = 5$ and $x_2^* = 0$, this becomes $5 + 0 \geq 4$, i.e. we have a *surplus* of 1 for the first constraint. Alternatively, we can get the amount of surplus by looking at the **coefficients** of the u_i in the objective row of the optimal tableau. Therefore, the amount of surplus for *constraint* 1 is the coefficient of u_1 in the *objective* row of the *optimal* tableau = 1. Similarly, constraint 2 has **no** surplus; constraint 3 has a surplus of **1**; and constraint 4 has **no** surplus.

The *shadow prices* for the i^{th} constraint of an LP problem is the amount by which the optimal value is *increased* (do not use the word "improved") if the right-hand side of the i^{th} constraint is *increased* by 1. The shadow prices are also the values of the **optimal** dual solutions, and thus are as *follows*: $u_1^* = 0$, $u_2^* = 110$, $u_3^* = 0$, and $u_4^* = 40$.

Interpretation. Consider that we change the *second* constraint from $x_1 + x_3 \geq 6$ to $x_1 + x_3 \geq 7$. This also changes the dual objective function from $z = 4u_1 + 6u_2 + 5u_3 + u_4$ to $z = 4u_1 + 7u_2 + 5u_3 + u_4$. Therefore, using the **optimal** values for the u_i , the optimal value changes *from* $Z_{OPT} = 4(0) + 6(110) + 5(0) + (40) = 700$ *to* $Z_{OPT} = 4(0) + 7(110) + 5(0) + (40) = 810$. There is a **change** of 110 above, therefore the shadow price for the *second* constraint (or for u_2) is **110**. Note that all of the above only applies if the *change in constraint i* leaves the current basis optimal.

24th February 2000

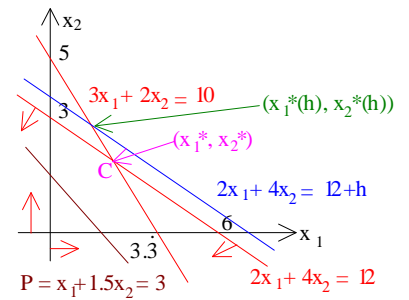
Back to the Paint Example

Reminder: *Primal* problem: Maximise $x_1 + \frac{3}{2}x_2$ subject to $2x_1 + 4x_2 \leq 12$ (u_1) and $3x_1 + 2x_2 \leq 10$ (u_2), with $x_1, x_2 \geq 0$. *Dual* problem: Minimise $12u_1 + 10u_2$ subject to $2u_1 + 3u_2 \geq 1$ (x_1) and $4u_1 + 2u_2 \geq 1.5$ (x_2), with $u_1, u_2 \geq 0$. We have the *optimal tableau* as shown on the right. Now let x_1^* , x_2^* , u_1^* and u_2^* be the *solutions* to the two problems. **Complementary slackness** gives $x_1^* = 0$ or $2u_1^* + 3u_2^* = 1$; $x_2^* = 0$ or $4u_1^* + 2u_2^* = 1.5$; $u_1^* = 0$ or $2x_1^* + 4x_2^* = 12$; and $u_2^* = 0$ or $3x_1^* + 2x_2^* = 10$. As we have $x_1^* = 2 = x_2^*$, $u_1^* = \frac{5}{16}$, and $u_2^* = \frac{1}{8}$, the *right hand* equations hold. A knowledge of x_1^* and x_2^* will give u_1^* and u_2^* (and vice versa).

1	0	-1/4	1/2	2
0	1	3/8	-1/4	2
0	0	5/16	1/8	5

Note that the **dual** optimal values are the *coefficients* of the corresponding slack variables in the expression for the primal objective function (at the **optimal** point). From our brief discussion on shadow "*prices*", we can link these coefficients with changes in the constraints. Suppose that the **pigment** constraint changes to $2x_1 + 4x_2 \leq 12 + h$.

Complementary slackness predicts that $2x_1^*(h)+4x_2^*(h) = 12+h$, and that $3x_1^*(h)+2x_2^*(h) = 10$. Therefore, $x_1^*(h) = 2^{-1/4}h$, and $x_2^*(h) = 2+^{3/8}h$, with the optimal value $5+^{5/16}h$. Note that the values $-^{1/4}$, $^{3/8}$ and $^{5/16}$ (the shadow prices for s_1) occur in the **tableau**. The solution will remain optimal for small values of h , but only in the **range** $x_1^*(h) \geq 0$ and $x_2^*(h) \geq 0$, i.e. $-^{16/3} \leq h \leq 8$. Similarly for changes to the **second** constraint: from 10 to $10+k$.



Changes in the Objective Function

For example, consider a change from M to $M(h) = x_1 + (^{3/2}+h)x_2$. We expect from the picture that this small change in the **gradient** of the objective function will not change the **position** of the optimal solution. Now $M(h)(2,2) = 5+2h$. For adjacent vertices, $M(h)(0,3) = ^{9/2}+3h$, and $M(h)(2,2) \geq M(h)(0,3)$, provided that $5+2h \geq ^{9/2}+3h$, i.e. $h \leq ^{1/2}$. Similarly, for the other near vertex, $h < 0$, and $(2,2)$ stays optimal provided that $h \geq ^{-5/6}$. The objective coefficient **range** for x_2 is $^{-5/6} \leq h \leq ^{1/2}$. For any h in this range, $(2,2)$ remains optimal.

Equality Constraints & Unrestricted Variables

If $2x_1+4x_2 = 12$ is given, replace it by *two* constraints: $2x_1+4x_2 \leq 12$, and $-2x_1-4x_2 \leq -12$. In the *dual*, we will get **two** variables u_1 and u_2 — but these will always occur together. For example, they *always* occur as (u_1-u_2) in the constraints: $2(u_1-u_2) \geq \dots$, $4(u_1-u_2) \geq \dots$; and in the **dual** objective function, they occur as $12(u_1-u_2)$. Let us define a *new* variable $u_3 = u_1-u_2$, so that u_3 is *unrestricted* (i.e. we do **not** have $u_3 \geq 0$). Dually, if we have an *unrestricted* variable in the primal problem, then we will have an **equality** constraint in the dual problem.

28th February 2000

A Transportation Problem

An *electrical utility* has 4 power stations, labelled $j = 1, 2, 3, 4$. These are supplied by three *collieries*, labelled $i = 1, 2, 3$. The total **requirement** of the power stations equals the total **supply** of coal; and the cost of transporting *one unit of coal* from each colliery to each power station is given in the table.

		j				Available Supplies
		1	2	3	4	
i	1	2	3	4	5	10
	2	5	4	3	1	15
	3	1	3	3	2	21
Requirements		6	11	17	12	

In the **corresponding** LP, x_{ij} is the amount transported from *colliery* i to *power station* j . So $x_{11}+x_{12}+x_{13}+x_{14} \leq 10$, $x_{21}+x_{22}+x_{23}+x_{24} \leq 15$, $x_{31}+x_{32}+x_{33}+x_{34} \leq 21$, and $x_{11}+x_{21}+x_{31} \geq 6$, etc. We have 7 constraints, where $7 = 3+4$, and they really are *equality* constraints. For the objective function, we want to minimise $2x_{11}+3x_{12}+4x_{13}+5x_{14}+\dots+2x_{34}$.

The LP problem has the *form* minimise $M = \sum_i \sum_j C_{ij}x_{ij}$ subject to $\sum_j x_{ij} = a_i$ (availables could have used \leq), with $\sum_i x_{ij} = b_j$ (requirements could have used \geq). In our case, the problem is *balanced*, in as much as $\sum_i a_i = \sum_j b_j$. Transportation method: (more efficient than simplex): (a) find a *basic feasible solution* (bfs); (b) **improve** on the bfs if possible.

Finding a bfs: North West Corner Method. Start at the *top left hand corner*, and make $x_{1,1}$ as large as possible, subject to the *constraints*. Delete the exhausted row (or column), with residual supply and requirements left. Repeat with the **new** problem. In the table shown, the total cost ($x_{11} = 6, x_{12} = 4, x_{22} = 7, x_{23} = 8, x_{33} = 9, x_{34} = 12$, all others zero) is $127 = (6 \times 2) + (4 \times 3) + (7 \times 4) + (8 \times 3) + (9 \times 3) + (12 \times 2)$.

	1	2	3	4	
1	6	4			10
2		7	8		15
3			9	12	21
	6	11	17	12	

2nd March 2000

Commentary

Allocation

	1	2	3	4	
1		10			10
2			3	12	15
3	6	1	14		21
	6	11	17	12	

2	3	4	5
5	4	3	2
1	2	3	2

$x_{1,1} = 6$ using up the *column* (it can't be more than 6 since $\sum x_{i,1} = 6$). Replace 10 by 4, the residual supply. In the **new** problem, $x_{12} = 4$, exhausting row 1. The *residual* requirement in column 2 is 7. $x_{2,2} = 7$. The residual supply in row 2 is 8; **delete** column 2 from the problem. $x_{2,3} = 8$. The residual requirement in column 3 is 9; *delete* row 2. $x_{3,3} = 9, x_{3,4} = 12$.

Note: An allocation can exhaust *row* and *column* constraints: this needs a special trick (see later). Total cost: 127.

Minimum Cost

Instead of heading for the *North West Corner* at each stage, head for the cell with the minimum cost. In the diagram shown, the total cost is $30 + 4 + 6 + 12 + 45 + 6 = 103$.

	1	2	3	4	
1		10			10
2		1	2	12	15
3	6			15	21
	6	11	17	12	

Vogel's Method

Look at the *relative unit cost* — this is the jump from the **lowest** entry to the **next** lowest entry in any row or column. In each row and column, note that the *difference* “(second smallest) - (smallest)” is the “jump”. Select a row or column with the largest jump. Allocate the maximum allowed to the cell in that row or column with the smallest unit cost.

6th March 2000

Improving on the Basic Feasible Solution

“Stepping Stones”. In the *first* diagram shown, we start by applying the North West Corner method, and end up with a total cost of 127. We see what a *reallocation* between cells

	6	4		10
	7	8		15
	9	12		21
R	6	11	17	12

would do. Try allocating θ to cell (1,3). The **change** in cost is $\theta(4 - 3 + 4 - 3) = 2\theta > 0$. This is not much use — it **increases** the cost. So we try again in the third diagram, with cell (2,4). Here, the *change* in cost is $\theta(1 - 3 + 3 - 2) = -\theta < 0$ (if $\theta \geq 0$). This could improve the solution — we could take $0 \leq \theta \leq 8$, so we could *decrease* the cost by -8.

6	4
7	8
	17 4

Doing this, we obtain the cell on the *left*, where the **total** cost is 119. We could do this for all empty cells, but with **m** supply points and **n** requirement points, there are mn cells in the table, and each solution will (usually) have $m+n-1$ positive values. Therefore, $mn - m - n + 1$ gets large very *quickly* — OOPS! So let us look again at the LP formulation **plus** its dual.

Primal: Minimise $M = \sum_i \sum_j c_{ij} x_{ij}$ subject to $\sum_j x_{ij} = a_i$ and $\sum_i x_{ij} = b_j$, with $x_{ij} \geq 0$ for all i and j . **Dual:** Maximise $N = \sum a_i u_i + \sum b_j v_j$ subject to $u_i + v_j \leq c_{ij}$ (u_i and v_j are unrestricted). The problem is *balanced* ($\sum a_i = \sum b_j$), and this means that the primal constraint **equations** are not linearly independent.

Lemma. At any *basic feasible solution*, the coefficient of x_{ij} in the primal objective function will be $c_{ij} - u_i - v_j$. Problem: to find *suitable* u_i and v_j . **Step 1:** Identify the *unit costs* for the (starting) position as shown on the right. **Step 2:** Attach **dual** variables u_i and v_j to rows and columns s.t. for each of these, $u_i + v_j = c_{ij}$. For our *example*, $u_1 + v_1 = 2$, $u_1 + v_2 = 3$, $u_2 + v_2 = 4$, $u_2 + v_3 = 3$, $u_3 + v_3 = 3$, and $u_3 + v_4 = 2$. We have 6 *equations in 7 unknowns*. Set one variable to a value of your choice ($v_1 = 0$). **Step 3:** For each cell, work **out** $c_{ij} - u_i - v_j$ as shown on the left.

2	3
4	3
	3 2

	v_1	v_2	v_3	v_4
$u_1 = 2$	2	3		
$u_2 = 3$		4	3	
$u_3 = 3$			3	2
	$=0$	$=1$	$=0$	$=-1$

0	0	2	4
2	0	0	-1
-2	-1	0	0

The $c_{ij} - u_i - v_j$ are the coefficients of the *objective function* at this point (at this bfs). To reduce the “cost”, head for the **maximum negative** entry (here, -2, in cell (1,3)) and allocate as much as possible. In the table, $\theta_{\max} = 6$, in order to keep non-negative allocations. The **total** change in cost is $\theta(1-2+3-4+3-3) = -2\theta$. So allocating $\theta = 6$ reduces the cost by 12. **Step 4:** allocate the maximum to this cell, and produce the *yellow* table. Then repeat until you get no $x_{ij} - u_i - v_j < 0$, and therefore you will obtain an optimal tableau.

6- θ	4+ θ		
	7- θ	8+ θ	
θ		9- θ	12
6	11	17	12

10			
15	10		
21	1	14	
6	6	3	12
6	11	17	12

8th March 2000

Tutorial: Transportation Problems

Q: For the *transportation* problem shown, use all three methods (**NW Corner, Minimum Cost and Vogel’s Method**) to find a starting position; (b) Use the *transport* algorithm to solve the problem (use one of the above starting positions). Note that in an exam, it is best to use the minimum cost, as using it with the algorithm usually ensures not too **long** a solution (NW Corner), nor not too **short** a solution (Vogel — doesn’t show that you *understand* the algorithm!).

	1	2	3	4	A
1	10	5	6	10	15
2	8	2	7	6	26
3	12	3	4	8	50
R	16	10	30	35	

NW Corner					
	1	2	3	4	A
1	15				15
2	1	10	15		26
3			15	35	50
R	16	10	30	35	

Minimum Cost					
	1	2	3	4	A
1	15				15
2		10			26
3	1		30	19	50
R	16	10	30	35	

A: Follow the algorithms to produce the *NW Corner* and the *minimum cost* solutions. Vogel’s method is not shown (see later!). Now apply the transportation algorithm using the **minimal** cost method to get the starting position.

	$v_1=0$	$v_2=-8$	$v_3=-8$	$v_4=-4$
$u_1=10$	10	5	6	10
	15			
	0	3	4	4
$u_2=10$	8	2	7	6
	θ	10		16- θ
	-2	0	5	0
$u_3=12$	12	5	4	8
	1- θ		30	19+ θ
	0	-1	0	0

Total Cost = 550. $\theta_{\max} = 1$.
New Total Cost = 548

	$v_1=0$	$v_2=-6$	$v_3=-6$	$v_4=-2$
$u_1=10$	10	5	6	10
	15			
	0	1	2	2
$u_2=8$	8	2	7	6
	1	10- θ		15+ θ
	0	0	5	0
$u_3=10$	12	5	4	8
		θ	30	20- θ
	2	-1	0	0

Total Cost = 548. $\theta_{\max} = 10$.
New Total Cost = 538

	$v_1=0$	$v_2=-7$	$v_3=-6$	$v_4=-2$
$u_1=10$	10	5	6	10
	15			
	0	2	2	2
$u_2=8$	8	2	7	6
	1			25
	0	1	5	0
$u_3=10$	12	5	4	8
		10	30	10
	2	0	0	0

Minimal Cost = 538

At the start, we get the v_i and the u_i by first saying that $v_1 = 0$, and getting the values **FROM** cells with values in them **USING** the unit cost matrix. For example, $u_1 = 10$, because $0 + ? = 10$ (10 from the **cost** matrix), so that $u_1 = 10$. The numbers in the *top left* denote the cost values, while the circled values denote which cells are used. The bottom right hand side values denote the *answer* of a cost value: $v_i - u_i$ for that particular cell. For example, cell (2,3) has value 5, which comes from $7 - (-8) - 10$.

Now, choose a cell with the **most negative** bottom right hand side coefficient; allocate θ to that cell; and do a “*circuit*” around the filled cells. $\theta_{\max} = 1$, because 1 is the *lowest* value on the circuit. So apply the change and proceed as above, until we obtain **no** negative bottom right hand side costs — and so we have an optimal solution: $x_{12} = 15$, $x_{21} = 1$, $x_{24} = 25$, $x_{32} = 10$, $x_{33} = 30$, $x_{34} = 10$, and $x_{ij} = 0$ otherwise.

Another Example: Here, we get a **degenerate** starting solution. We can ignore it, and declare that when we fixed an *allocation* of $x_{12} = 20$, that “killed” the first row. To complete the table, we thus put a 0 in the 2nd column, $x_{22} = 0$, to complete the column artificially. Alternatively, if we consider that 20 was filling the 2nd column, we then add in a zero somewhere in the **first** row.

Unit Cost Matrix	A	UCD	Vogel's Method
2 1 4 5	20	1	20
5 4 3 5	10	λ 2	10
6 2 1 3	35	λ 5 ←	8
3 6 2 8	16	1	2
R 14 20 20 27			14
UCD λ 1 1 2		UCD = Unit Cost Difference	
2 ←			

This is a bit ad hoc — we used the *perturbation* method to solve the problem. Add ϵ to each of the “available supplies”, A, and collect up all these notational supplies with an additional *requirement* of 4ϵ in the first column. Then, use Vogel (or any other method) to solve the problem. In the solution table, we select which cell gives us a “zero” basic variable. It is optimal, as we have no negative coefficients; and therefore the minimal total cost is $185 + 11\epsilon$. **Now** put $\epsilon = 0$, and the *minimum* total cost is 185 (at $x_{12} = 20$, $x_{23} = 10$, $x_{33} = 8$, $x_{34} = 27$, $x_{41} = 14$, $x_{43} = 2$, and all the others zero). Note that $c_{24} = u_2 + v_4$, so this solution is not *unique*.

	$v_1=0$	$v_2=-1$	$v_3=-1$	$v_4=1$	
$u_1=2$	② ϵ	① 20	4	5	20+ ϵ
	0	0	3	2	
$u_2=4$	5	4	③ 10+ ϵ	5	10+ ϵ
	1	1	0	0	
$u_3=3$	6	2	④ 8+ ϵ	3	35+ ϵ
	4	1	0	0	
$u_4=3$	③ 14+3 ϵ	6	② 2-2 ϵ	8	16+ ϵ
	0	4	0	4	
	14+4 ϵ	20	20	27	

9th March 2000

Compact Notation

Use **bigger** cells to show all the *information* clearly. Circle those c_{ij} with positive x_{ij} , and put the largest -ve bottom right hand corner in a little square. Remember to add comments, e.g. “Total cost = 127”, “ $\theta_{\max} = 6$ ”, “Allocate 6 to (1,3)”, and “Note: **choice** of -1 as the pivot cell”. At the end, say “*No negative bottom right hand corners, so this tableau is optimal*”, and list the coefficients, remembering those you do not list are zero. If possible, use different **colours** for clarity.

c_{ij}
$x_{ij} (\pm\theta)$
$c_{ij} - u_i - v_j$

Assignment 3: Set 8/3; In 17/3; Back 24/3

Q: For the *transportation problem shown*, (a) Use all of the **three** methods (NW Corner, Minimum Cost and Vogel’s Method) to find a **starting** position; (b) use the transport algorithm to **solve** the problem. A: (see over).

	1	2	3	4	A
1	22	11	4	7	5
2	6	10	7	5	9
3	19	8	5	2	15
R	3	5	9	12	

For the problem, which is *balanced*, we now find basic feasible solutions. (a) Using the **NW Corner** method, we obtain the diagram on the right, with total cost 199. *Commentary:* We allocate 3 units to the top left, because this is the maximum we can allocate without exceeding the requirements or the availables for that row/column. This exhausts the **requirements** for the first column, and reduces the availables for the first row to 2.

3	2			
3	6			
		3	12	

We now exhaust the *availables* for the first row, by allocating 2 units to cell (1,2), checking first that the requirement for the second column at least matches 2. Now repeat the above, starting with cell (2,2). (b) Using the *minimum cost* method, we end up with a b.f.s. with total cost 134. *Commentary:* Start with cell (3,4), because this has the lowest unit cost, 2. Allocate 12 to this cell, then hide the fourth column, and start on the second iteration, etc.

3	5	1		
3	6			
		3	12	

22	11	4		
6	10	7		
19	8	5		

(c) Using **Vogel's** method, we get a *basic feasible solution* with a total cost of 134 units. *Commentary:* start off by calculating the **relative costs**, shown in the matrix headed "1st iteration". The biggest jump occurs in the 1st column (from 6 to 19), so we allocate as much as we can to cell (2,1), which is 3 units. Now ignore the first column, and start on the second iteration. Here, there are **three** choices for the largest jump. I decided to choose the cell for which we could allocate the most units to. So I allocated 12 units to cell (3,4), rather than 9 units to cell (1,4). Now *carry on*....

3	5			
3	4			
3	12			

22	11	4	7	3
6	10	7	5	1
19	8	5	2	3
				UCD

11	4	7	3	5
10	7	5	1	
8	5	3	12	
2	1	3		UCD

11	4	7	3	
10	7	3		
8	5	3	5	3
2	1			UCD

I will now use the **transport** algorithm with the b.f.s. given by the minimum cost method. Constructing the "*information matrix*", we get what is shown, where all the symbols and locations have their usual meanings. Because there are no negative bottom right hand corners, the tableau is optimal, and so we have a minimum cost of 134, with the following optimal values: $x_{13} = 5$, $x_{21} = 3$, $x_{22} = 5$, $x_{23} = 1$, $x_{33} = 3$, $x_{34} = 12$, and all other $x_{ij} = 0$. The red circled zero shows that the optimal solution is not *unique*. (If the bottom right's weren't **all** non-negative, we'd allocate θ units to the *most* negative, and go on a circuit to balance things out. The total cost then **reduces** by $\theta \times (\text{bottom right hand corner concerned})$).

13th March 2000

Special Situations

(i) Unbalanced transportation problems. If the total *supply* is not equal to the total *requirements*, the problem is unbalanced, and either surplus will be **stored**, or slack **bought in** from outside the system **Example** (A variant on the basic problem): In the *first* table, the total available supply is 49, and the total requirement is 46.

	1	2	3	4	A
1	2	3	4	5	11
2	5	4	3	1	16
3	1	3	3	1	22
R	6	11	17	12	surplus
	1	2	3	4	A
1	2	3	4	5	p1 11
2	5	4	3	1	p2 16
3	1	3	3	1	p3 22
R	6	11	17	12	3 dummy
	1	2	3	4	A
1	2	3	4	5	p1 a1
2	5	4	3	1	p2 a2
3	1	3	3	1	p3 a3
dummy	q1	q2	q3	q4	0 a4
R	b1	b2	b3	b4	b5

In the *second* tableau, the p_i 's are penalty costs, perhaps of *transportation, storage, repayment of loans*, etc. For simplicity, assume that the p_i 's are **zero** if not told otherwise. Then, run the transport algorithm as normal. If need be, add a dummy row as well as a dummy column. This is shown in the third tableau, where a_4 is a *dummy supply level*, and b_5 is a *dummy requirement level*. We **need** $\sum a_i = \sum b_j$, and can take a_4 and b_5 as *large* as we like to ensure that this can happen. The **usual** way is to take $a_4 = b_1 + b_2 + b_3 + b_4$, and to take $b_5 = a_1 + a_2 + a_3$.

(ii) Non-unique solutions. In a normal LP problem, *non-unique* solutions occur when M is constant along some face of the feasible region. In this situation, at a **vertex** on that region, we have some *non-basic* variables b_1, \dots, b_m which are equal to zero at that vertex, and some *basic* variables b_{m+1}, \dots, b_n which correspond to a **column** having zeros and a single 1.

If we include a *non-basic* variable b_l , we expect the *objective* function M to change by $\Delta b_l \times m_l$, where m_l is the *coefficient* of M in column l . If we are at an **optimum**, yet that optimum is not unique, then we must have $m_l = 0$, and vice versa. We thus have: if the coefficient of a non-basic variable in the *optimum* function M is zero, then if we are at an optimum, that optimum is not unique.

Back to **transportation** problems. For the diagram shown, the bottom right hand corner is $c_{ij} - u_i - v_j$ in each cell. If it happens that this entry is *zero* for a non-allocation cell in an optimal tableau, then the optimal solution is not unique. **Example:** Look at the diagram on the right. We end up with a *total cost* of 102. But note: $c_{13} - u_1 - v_3 = 0$. So allocating θ to cell (1,3) *changes* the cost by $(3 - 3 + 3 - 3)\theta$, i.e. not at all. **Any** θ , for $0 \leq \theta \leq 10$, could be *allocated* to (1,3) **without** changing costs.

2	3	3	4	10
5	4	3	1	15
1	3	3	2	21
6	11	17	12	

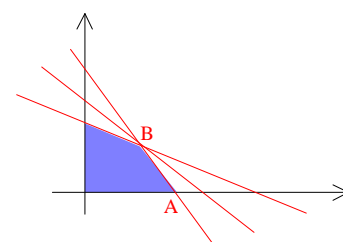
2	3	3	5
1	10- θ	0	4
5	4	3	12
4	1	3	0
1	3	3	2
6	1+ θ	14- θ	1
0	0	0	0

20th March 2000

Degeneracy (Not Examinable)

b.v.	x_1	...	x_{is}	...	x_j	...	x_{n+m}	
x_r	a_{r1}		0		a_{rj}		$a_{r,n+m}$	c_r
x_s	a_{s1}		1		a_{sj}		$a_{s,n+m}$	c_s
...

In L.P. problems, the situation shown in the *diagram* can very, very occasionally occur. In the tableau, suppose that we have what is shown. **Notes:** b.v. = *basic variable*; the two values c_r and c_s are such that $c_r/a_{rj} = c_s/a_{sj}$; and the x_j column is the *pivot* column.



Pick **one** of the two rows as the pivot row. We will pick the r^{th} row, and reduce to get the tableau *shown on the right*. So the *new basic variable* x_j has value c_r/a_{rj} , but the basic variable x_{is} has become 0 during the process. With this situation, the algorithm may *loop infinitely*. If it does, kick it by slightly **perturbing** the problem.

	x_1	...	x_{is}	...	x_j	...	x_{n+m}	
x_r	a_{r1}/a_{rj}		0		1		$a_{r,n+m}/a_{rj}$	c_r/a_{rj}
x_s			1		0		$a_{s,n+m} - \dots$	$c_s - a_{sj}(c_r/a_{rj}) = 0$
...

Back to Transportation Problems

Here, degeneracy corresponds to *one of the $m+n-1$* allocations of our solution being zero. For example, in the problem shown, if we use the NW Corner method, we get the **second** matrix. Here, there are 5 (rather than 6) non-zero allocations. So, before continuing, set one of the remaining allocations to *zero*. If the algorithm loops, apply a small perturbation: with $\epsilon > 0$, we have the *bfs* as shown on the left. How **big** should ϵ be? If we change the a_i to $a_i + \epsilon$, and the b_1 to $b_1 + m\epsilon$, and the other levels of supply remain *unchanged*, using $\epsilon = 1/2n$ usually **works**.

2	3	4	5	10
5	4	3	1	15
1	3	3	1	21
10	11	11	14	

10	0			$\cancel{10}$
	11	4		$\cancel{15}$
		7	14	$\cancel{21}$

$10+\epsilon$				
2ϵ	11	$4-\epsilon$		
		$7+\epsilon$	14	
$10+3\epsilon$	11	11	14	

the b_1 to $b_1 + m\epsilon$, and the other levels of supply remain *unchanged*, using $\epsilon = 1/2n$ usually **works**.

22nd March 2000

Johnson's Algorithm for the Two Machine Flow Shop

Suppose that we have *two machines*, and jobs have to be processed first on M_1 , and then on M_2 . Because the jobs “flow” between the machines, and we are optimising the processing, we have a flow shop problem. Let the processing time of job J_i on machine M_1 be α_i , and that on M_2 be β_i . We seek to *minimise* the overall time to complete the whole job, the **makespan**.

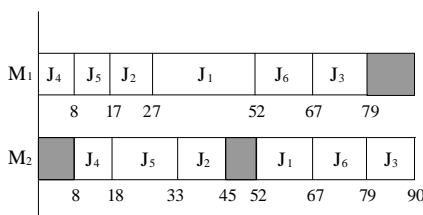
Johnson's Algorithm. (1) Select the *smallest* time in the list $\alpha_1, \alpha_2, \dots, \alpha_n, \beta_1, \beta_2, \dots, \beta_n$. If there is a tie, select *any* of the smallest times. (2) If the smallest time is an α_k , do the k^{th} job first. If it is a β_k , do the k^{th} job last. (3) Now repeat the procedure on the *remaining* $(n-1)$ jobs.

Johnson's Algorithm for the Two Machine "Job Shop"

Here, the **order** of processing (M_1, M_2) is not always the same. We have 4 classes of jobs: N_1 : *processing* M_1 then M_2 ; N_2 : *processing* M_2 then M_1 ; N_3 : only needing M_1 ; and N_4 : only needing M_2 . To find the *optimal* schedule, do the following: (1) Schedule the jobs in N_1 (with the flow shop algorithm) to get the **sequence** S_1 . (2) Schedule the jobs in N_2 to get S_2 , remembering that M_2 is the *first* machine. (3) Schedule the jobs in N_3 in **any** order to get S_3 ; (4) Schedule the jobs in N_4 in *any order* to get S_4 . An *optimal* schedule is then: process the jobs on M_1 in the order (S_1, S_3, S_2) ; process the jobs on M_2 in the order (S_2, S_4, S_1) .

Exercise 1: Use Johnson's algorithm to find the *optimal* schedule for the shown set of jobs. A: The smallest time is α_4 , so do job J_4 first. Next, we have $\alpha_5 = 9$, so do job J_5 second — the list so far is (J_4, J_5, \dots) ; next $\alpha_2 = 10$ so (J_4, J_5, J_2, \dots) ; next $\beta_3 = 11$ so $(J_4, J_5, J_2, \dots, J_3)$; next $\beta_6 = 12$ so $(J_4, J_5, J_2, \dots, J_6, J_3)$; finally put the last in the *middle*, so we have an **optimal** schedule $(J_4, J_5, J_2, J_1, J_6, J_3)$.

J_i	α_i	β_i
1	25	15
2	10	12
3	12	11
4	8	10
5	9	15
6	15	12



To get the *optimal time*, draw a **Gantt** diagram like the one shown on the **left**, and find out that the *makespan* is 90, with an idle time of 11 for M_1 and 15 for M_2 . These are *unavoidable*.

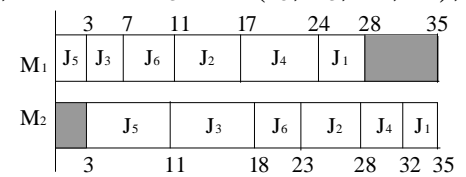
Exercise 2: Use Johnson's algorithm to find the optimal schedule for the set of jobs in the "job-shop" shown. Here, the schedule for N_1 (in Red) is $S_1 = J_1, J_{10}, J_{11}, J_6$; and the schedule for N_2 (in Blue) is $S_2 = J_9, J_3, J_5, J_8$. Always be careful with this one (with the ordering). The schedule for S_3 (in Purple) is J_2, J_{12} ; and the schedule for S_4 (in Green) is J_4, J_7 . Now draw a diagram similar to the one above, remembering that the ordering is (S_1, S_3, S_2) for M_1 , and (S_2, S_4, S_1) for M_2 . We find out that the total time is 168, with an idle time of 36 for machine 2.

Job	1	2	3	4	5	6	7	8	9	10	11	12
α_i	20	17	21		17	15		20	10	15	16	17
β_i	14		11	10	14	10	15	25	10	12	11	
Group	1		2		2	1		2	2	1	1	

Assignment 4: Set 23/3; In 31/3; Back 7/4

J_i	1	2	3	4	5	6
α_i	4	6	4	7	3	4
β_i	3	5	7	4	8	5

Q: Use Johnson's algorithm to find the optimal schedule for the shown set of jobs. Give the total optimal makespan, and the idle time on each machine. A: Choose $\alpha_5 = 3$ (J_5, \dots); Next $\beta_1 = 3$ (J_5, \dots, J_1); Choose $\alpha_3 = 4$ (J_5, J_3, \dots, J_1); Choose $\alpha_6 = 4$ ($J_5, J_3, J_6, \dots, J_1$); Next $\beta_4 = 4$ ($J_5, J_3, J_6, \dots, J_4, J_1$). We only have job J_2 left, so the optimal schedule is $(J_5, J_3, J_6, J_2, J_4, J_1)$. The Gantt diagram for this schedule is as shown on the right. Optimal Makespan: 35 units. Idle Time: Machine 1: 7 units; Machine 2: 3 units.



J_i	1	2	3	4	5	6	7	8	9	10	11	12	13	14
α_i	20	17	21		7	5		24	10	16	16	17	15	15
β_i	14		11	10	14	10	15	25	10	11	11		12	12
G	1		2		2	1		2	2	1	1		1	3

Q: Use Johnson's algorithm to find the optimal schedule for the shown set of jobs in a "job-shop". Give the optimal schedule, the makespan, and any idle time. Give a Gantt diagram to show that your schedule is feasible. Comment on any flexibility in the schedules, for example if the positions of two jobs can be exchanged without adversely affecting the makespan.

J_i	α_i	β_i	J_i	α_i	β_i	J_i	α_i	β_i	J_i	α_i	β_i
J_1	20	14	J_3	21	11	J_2	17		J_4		10
J_6	5	10	J_5	7	14	J_{12}	17		J_7		15
J_{10}	16	11	J_8	24	25						
J_{11}	16	11	J_9	10	10						
J_{13}	15	12	J_{14}	15	12						

For Group 1, we have the following schedule (from the information in the first table): $(J_6, J_1, J_{13}, J_{11}, J_{10}) = S_1$. For Group 2, $S_2 = (J_3, J_{14}, J_8, J_9, J_5)$. For Group 3, $S_3 = (J_2, J_{12})$ (scheduling in any order). And for

Group 4, $S_4 = (J_4, J_7)$ (scheduling in any order). Machine 1 processes in the order S_1, S_3, S_2 — so has schedule $(J_6, J_1, J_{13}, J_{11}, J_{10}, J_2, J_{12}, J_3, J_{14}, J_8, J_9, J_5)$. Machine 2 processes in the order S_2, S_4, S_1 — so has schedule $(J_3, J_{14}, J_8, J_9, J_5, J_4, J_7, J_6, J_1, J_{13}, J_{11}, J_{10})$.

Forming a Gantt diagram for this problem, we would find an optimal makespan of 183 units, and an idle time of 0 units for machine 1, and 28 units for machine 2. There is a lot of flexibility in the schedules. We can interchange the order of any two jobs in fact; and with suitable restrictions, it is also possible to change the order of the sequences.

For example, we can interchange the order we do S_1 and S_3 on machine 1, with jobs in these sequences being able to be shuffled in any way, as long as job 6 is not the last job to be done.

Scheduling on a Single Machine

Characteristics: There are n jobs to be *processed* on a single machine. **Parameters & Variables:** *Data:* p_i is the *processing time* for job J_i ; d_i is the *due date* for job J_i . *Dependent on Schedule:* C_i is the *completion time* for job J_i ; L_i is the *lateness* of job J_i ($L_i = C_i - d_i$); T_i is the *tardiness* of job J_i ($T_i = \max(L_i, 0)$). *Optional:* **Weight** factors w_i , giving the relevant *importance* of the jobs. **Problems:** To minimise $\bar{C} = \frac{1}{n} \sum_{i=1}^n C_i$, the average *completion time*; or $C_{\max} = \max_{i=1}^n \{C_i\}$, the *makespan*; or \bar{L} , the average *lateness*; or L_{\max} ; or **prioritised** versions, e.g. $\frac{1}{n} \sum w_i C_i$.

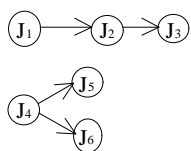
(i) **SPT Scheduling** (*Shortest Processing Time*). We have *equally* weighted jobs, and want to *minimise* \bar{C} . **Algorithm:** Sequence the jobs so that job J_i is processed before job J_j if $p_i < p_j$. (If $p_i = p_j$, toss a coin!). **Example:** the table shown, where the due date is now! The schedule is $J_4, J_5, J_6, J_2, J_3, J_1$, where J_5 & J_6 , and J_2 & J_3 can be **permuted**.

J_i	1	2	3	4	5	6
p_i	20	15	15	5	10	10

Work out the *completion times* as shown on the right, then $\bar{C} = \frac{215}{6} = 35\frac{5}{6}$. Why does this *work*? If the schedule is $J_{i_1}, \dots, J_{i_k}, \dots, J_{i_n}$, then $C_{i_1} = p_{i_1}$; $C_{i_2} = p_{i_1} + p_{i_2}$; ...; $C_{i_n} = p_{i_1} + \dots + p_{i_n}$; and $\sum C_i = np_{i_1} + (n-1)p_{i_2} + \dots + 2p_{i_{n-1}} + p_{i_n}$. We *minimise* this by keeping the **longer** jobs until later. To *minimise* \bar{L} , $L_i = C_i - d_i$; and $\frac{1}{n} \sum L_i = \frac{1}{n} \sum C_i - \frac{1}{n} \sum d_i$; so that $\bar{L} = \bar{C} - \bar{d}$, where \bar{d} is the *average due date*, **independent** of the schedule. So to *minimise* \bar{L} , *minimise* \bar{C} .

J_i	4	5	6	2	3	1
C_i	5	15	25	40	55	75

(ii) **EDD Scheduling** (*Earliest Due Date*). To *minimise* L_{\max} or T_{\max} , process job J_i before job J_j if $d_i < d_j$. (iii) **Lawler's Algorithm**. We have *non-equal* priority weightings, and *precedence* constraints. In the example shown (with accompanying diagram), we want to **minimise** the maximum tardiness, $T_{\max}^w = \max_{i=1}^n \{w_i T_i\}$.



J_i	p_i	d_i	Prec.	w_i
1	2	3	-	1
2	3	6	J_1	1
3	4	8	J_2	1
4	3	7	-	2
5	2	12	J_4	2
6	1	7	J_4	2

Lawler's Algorithm: find the job to schedule **last**, then the one to schedule last but **one**, etc. Let V be the set of jobs that can be performed last. (In the example, $V = \{J_3, J_5, J_6\}$). Let $\tau = \sum_{i=1}^n p_i$. Choose a job in V to *minimise* $w_i \max\{\tau - d_i, 0\}$. Suppose that J_k does this, then schedule J_k **last**. Remove J_k from the list and repeat.

Tutorial

Let the aim be to *minimise* the **maximum** weighted tardiness, $T_{\max}^w = \max_{i=1}^n \{w_i T_i\}$. An example table is shown on the *right*, where P denotes precedence (the previous jobs). Let V be the set of jobs that can be performed **last**, $V = \{J_3, J_5, J_6\}$. Let $\tau = \sum_{i=1}^n p_i$ be the *total completion time* of the last job. Note: τ does not depend on the **order** in which the jobs are processed.

J_i	p_i	d_i	P	w_i
J_1	2	3	-	1
J_2	3	6	J_1	1
J_3	4	8	J_2	1
J_4	3	7	-	2
J_5	2	12	J_4	2
J_6	1	7	J_4	2

Choose a job in V which *minimises* $w_i \{\tau - d_i, 0\}$. Suppose that J_k does this, so put J_k last; **remove** J_k to obtain a *new* problem with $(n-1)$ jobs; and repeat the procedure.

τ	J_i	J_1	J_2	J_3	J_4	J_5	J_6	SJ
	p_i	2	3	4	3	2	1	
	d_i	3	6	8	7	12	7	
	w_i	1	1	1	2	2	2	
15		*	*	7	*	6	16	J_5
13		*	*	5	*	S	12	J_3
9		*	3	S	*	S	4	J_2
6		3	S	S	*	S	0	J_6
5		2	S	S	0	S	S	J_4
3		0	S	S	S	S	S	J_1

An example of tabulated calculation is as shown on the left. **Key:** SJ = *Scheduled Jobs*; * indicates that this job cannot be *scheduled* in the current last position due to precedence rules; S indicates that the job has already been **scheduled** (helps with the “book-keeping”); and **red** indicates choices. After finishing, the *optimal* schedule is ($J_1, J_4, J_6, J_2, J_3, J_5$), with a *weighted tardiness* of 6 (the maximum **red** element).

Another example of calculation is as shown on the right. Note that it is important that you keep in mind which jobs are **completed** as you go along — some precedence will be lifted.

τ	J_i	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8	SJ
	p_i	2	3	5	3	2	2	6	1	
	d_i	5	6	8	13	12	10	7		
	w_i	1	1	3	2	2	3	2	2	
24		*	*	48	*	24	36	28	34	J_5
22		*	*	42	*	S	30	24	30	J_7
16		*	*	24	*	S	12	S	18	J_6
14		*	*	18	*	S	S	S	14	J_8
13		*	*	15	0	S	S	S	S	J_4
10		*	*	6	S	S	S	S	S	J_3
5		*	0	S	S	S	S	S	S	J_2
2		0	S	S	S	S	S	S	S	J_1

Smith’s Algorithm (*a preview*): Suppose that the weightings are **equal**, and no precedence rules apply. Minimise \bar{C} subject to no job being *late*. The algorithm is adapted from the above, but you are not allowed to have a job completed **after** its due date.

Remark: If the weighting ($w_i T_i$) is replaced by a *non-decreasing* function of T_i or of C_i , then the algorithm **still** works. (Each function would in general be *different*). In our case, $\gamma_i(x) = w_i \max\{x - d_i, 0\}$, where x is the *completion* time C_i .

(iv) **Smith’s Algorithm.** Goal: to minimise $\bar{C} = \frac{1}{n} \sum_{i=1}^n C_i$ subject to *no job being late* — $T_i = 0$ for every i . An example calculation is shown on the **right**. Note that $\tau = \sum p_i$ again, and that V is the set of jobs that can be *scheduled* last without being late. Select from V a job J_k with p_k as large as possible. Put J_k last, delete from the *problem* and repeat. For the problem shown, $\bar{C} = \frac{1}{6}(1+2+6+8+10+13) = \frac{40}{6} = 6\frac{2}{3}$; and we reach this minimum with the schedules ($J_6, J_2, J_3, J_5, J_1, J_4$) or ($J_2, J_6, J_3, J_5, J_1, J_4$).

τ	J_i	J_1	J_2	J_3	J_4	J_5	J_6	SJ
	p_i	2	1	4	3	2	1	
	d_i	10	15	7	14	9	11	
13		*	1	*	3	*	*	J_4
10		2	1	*	S	*	1	J_1
8		S	1	*	S	2	1	J_5
6		S	1	4	S	S	1	J_3
2		S	1	S	S	S	1	J_2
1		S	S	S	S	S	1	J_6

Multi-Machine Processes

Consider Johnson’s algorithm for the *3 machine flow shop* with machines M_1, M_2 and M_3 ; and let the times for job J_i be a_i, b_i and c_i for each machine respectively. **Minimise** the makespan, assuming that the b_i are “*dominated*”, i.e. either $\min_{1 \leq i \leq n} a_i \geq \max_{1 \leq i \leq n} b_i$, or $\min_{1 \leq i \leq n} c_i \geq \max_{1 \leq i \leq n} b_i$. **Method:** Consider the *two machine* flow shop problem with $\alpha_i = a_i + b_i$, and $\beta_i = b_i + c_i$.

3rd April 2000

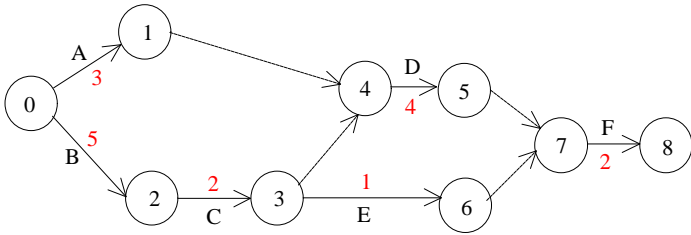
For the example shown on the right, the domination condition holds, with $\min a_i = 5 \geq \max b_i = 4 \leq \min c_i = 5$. **Johnson’s** Algorithm gives the schedule ($J_1, J_6, J_4, J_5, J_2, J_3$). Now draw a *triple Gantt diagram* as shown on the left, where the makespan is 51.

Job	a_i	b_i	c_i	α_i	β_i
J_1	5	2	6	7	8
J_2	7	4	5	11	9
J_3	9	1	6	10	7
J_4	6	4	8	10	12
J_5	12	3	9	15	12
J_6	5	2	7	7	9

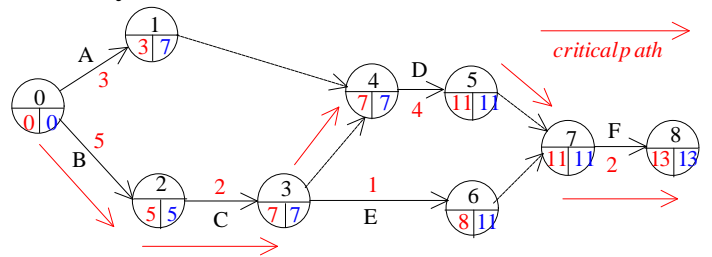
Critical Paths

When the *number of activities* is larger, they are nearly always interlinked. Consider the table on the right. Drawing a network diagram to illustrate the **dependence** relations, the nodes of the network mark the beginning and end of each task, while the edges themselves represent the **tasks**. The dotted lines represent the “*dummy*” activities, while the **red** numbers denote the “*durations*” of each activity.

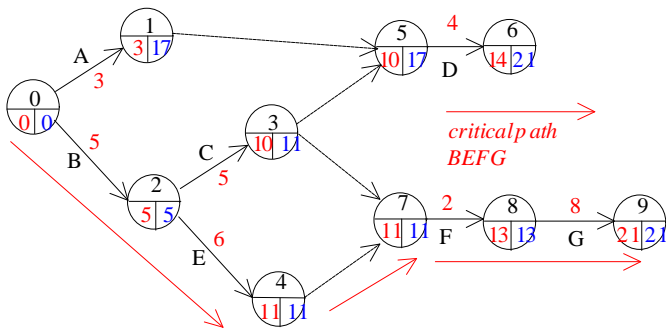
Activity	A	B	C	D	E	F
Depends On	-	-	B	A, C	C	D, E



Now replace each node by a subdivided **disc**. Key: bottom left = e_i , the **earliest** time at node i ; bottom right = l_i , the **latest** time at node i . Work back *from the end* to get the latest times l_i , so that each node can be reached **without** delaying the completion time. The critical path is the path through the network taking the **longest** time. On the critical path, $e_i = l_i$, i.e. there is no time to spare.



In the example shown above, the total time **sequentially** will be at least $13+3+1 = 17$ days. If resources allow the tasks to be done in “*parallel*”, then we can complete the tasks in 13 days. For a delay on a non-critical activity, the completion time for parallel execution will not change (for small delays). In the above example, the critical path is BCDF. To check that two teams *can* perform all the tasks, draw a **Gantt** diagram.

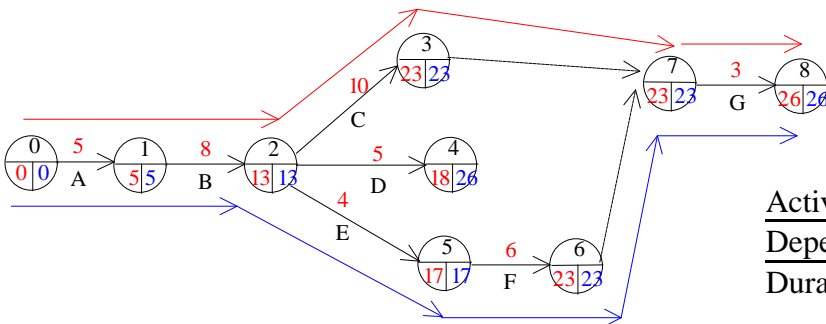


Activity	A	B	C	D	E	F	G
Depends on	-	-	B	A, C	B	C, E	F
Duration	3	5	5	4	6	2	8

Consider *another* example, with the table shown above, and the diagram shown on the left.

5th April 2000

Q: Find the **critical** path(s) for the example shown. A: The *critical* paths are ABEFG and/or ABCG.



Activity	A	B	C	D	E	F	G
Depends on	-	A	B	B	B	E	C, F
Duration	5	8	10	5	4	6	3

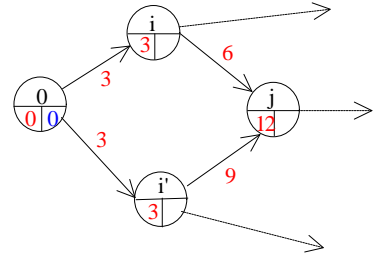
6th April 2000

Note: Each activity corresponds to an arc *from a start node i to a terminal node j* . It is often useful to call the arc or the activity by the **pair** (i, j) .

Further Analysis

The **total float** on an arc (i, j) is the amount by which the *starting* time of (i, j) could be delayed beyond its earliest starting time — without delaying the completion of the project. So $TF(i, j) = l_j - e_i - t_{ij}$, where t_{ij} is the duration of activity (i, j) . Activities on the *critical* path have total float 0. These are called the critical **activities**.

The free float of an arc (i, j) is defined as $FF(i, j) = e_j - e_i - t_{ij}$. This interpretes as the *amount* by which the starting time of activity (i, j) can be delayed — without delaying the start of any later activity beyond its earliest starting time. This can be non-zero. Suppose we have a **diagram** as shown, then $FF(i, j) = 12 - 3 - 6 = 3$.



Formulation of a Critical Path Problem as a Linear Programming Problem

Let x_j be the time at which *node j occurs*; and let t_{ij} be the duration of activity (i, j) . Therefore, $x_j \geq x_i + t_{ij}$. If F is the **final** node of the project (and if one doesn't exist naturally, *add* dummy arcs to a new final node), then the completion time of the project is $x_F - x_0$. **Example** (using the first C.P. diagram): each arc yields a *constraint*, summarised as follows:

Minimise $x_8 - x_0$ subject to: $x_1 \geq x_0 + 3$ (arc $(0,1)$); $x_2 \geq x_0 + 5$ (arc $(0,2)$); $x_3 \geq x_2 + 2$ (arc $(2,3)$); $x_4 \geq x_1$; $x_4 \geq x_3$ (*dummies*); $x_5 \geq x_4 + 4$ (arc $(4,5)$); $x_6 \geq x_3 + 1$ (arc $(3,6)$); $x_7 \geq x_6$; $x_7 \geq x_5$ (*dummies*); $x_8 \geq x_7 + 2$ (arc $(7,8)$), with $x_i \geq 0$. We can **read off** an optimal solution $x_i = e_i$, with optimal *time* 13.

Sensitivity Analysis

Writing the constraints in a more *standard* form, for example $x_5 - x_4 \geq 4$, we can use **sensitivity analysis** to see how the completion time varies if 4 is varied *slightly*.

Crashing the Project

The above example completes in 17 days if done *sequentially*, and in 13 days if done in *parallel*. Suppose a bonus would be paid if the completion time was **less** than 10 days — but extra resources *cost*. Let C_{ij} be the cost of reducing task (i, j) by **one** day, and let r_{ij} be the amount we reduce task (i, j) by. The cost of *crashing* the project is therefore $\sum_{\text{arc}} C_{ij} r_{ij}$.

Take this as the *new objective function*. Then take a new **constraint**: $x_F - x_0 \leq 10$; and change the *other* constraints: arc $(0,1)$ to $x_1 \geq x_0 + 3 - r_{01}$; arc $(0,2)$ to $x_2 \geq x_0 + 5 - r_{02}$; etc. *Superficial* analysis shows that whatever the C_{ij} 's are, we do not want to **reduce** non-critical activities unless the "*range of validity*" of our critical path (the optimal solution) is **exceeded**.

Exam Paper: May 2000

SECTION 1 (Compulsory)

- (1) A toy manufacturer produces two types of toy, A and B. When running at full capacity, the factory has at most 1000 man-hours available in its machining department, 400 man-hours available in its assembly department and 250 man-hours in its painting department. Manufacturing time (in hours) required by each toy in the three departments is given by:

	Machining	Assembly	Painting
A	0.2	0.2	0.1
B	0.5	0.1	0.1

A toy of type A sells for £10 and one of type B for £8.

- (a) Formulate a linear program model of this situation, explicitly defining the variables, the constraints they must satisfy and the objective function. **[8 marks]**
- (b) Use the simplex algorithm to find the production levels that will maximise the income. **[12 marks]**

SECTION 2 (Answer 2 out of 4 questions)

- (2) Consider the following minimisation problem:

$$\begin{aligned} \text{Minimise } w &= 4y_1 + 3y_2 + y_3 \\ \text{subject to } y_1 + 2y_2 &\geq 6 & (1) \\ y_1 - y_2 + y_3 &\geq 8 & (2) \\ 2y_2 + y_3 &\geq 10 & (3) \end{aligned}$$

where $y_1, y_2, y_3 \geq 0$.

Find the dual problem, solve the dual problem and form the dual optimal tableau, find the solution to the primal problem. State the amount of surplus in each primal constraint and identify and interpret the shadow prices. **[15 marks]**

- (3) A small manufacturing group has 3 warehouses and 4 retail outlets. The relative costs of transporting one item from warehouse to outlet are given in the following table, as are the resources available (right hand column) and the supply requirements (bottom row):

Outlets:	1	2	3	4	Resources
Warehouse 1	20	15	5	6	5
Warehouse 2	5	12	10	2	10
Warehouse 3	15	7	4	1	15
Requirements:	4	4	10	12	

- (a) Use the NW corner, the minimum cost **and** Vogel's method to find basic feasible solutions as starting positions for the transport algorithm solution of this problem. **[6 marks]**
- (b) Solve to find the optimal assignment, starting with the minimum cost basic feasible solution. **[9 marks]**
- (4) (a) The processing times for five jobs on each of two machines is shown in the table below:

Job	Machine 1	Machine 2
1	4	3
2	2	2
3	5	4
4	3	5
5	4	6

All jobs have to be processed first on Machine 1 before passing to Machine 2. Find the optimal order so as to minimise makespan and draw a Gantt diagram for your schedule giving the total time taken, and the total idle time. **[7 marks]**

- (b) Seven jobs have to be processed on a single machine. Job, J_k , takes p_k days to process and has due date d_k . The various values of p_k and d_k are given in the following table:

J_k	J_1	J_2	J_3	J_4	J_5	J_6	J_7
p_k	1	2	3	4	1	3	3
d_k	10	8	16	6	9	17	13

Let C_k be the completion time for J_k and set $\bar{C} = (\sum C_k)/7$, the average completion time. Using Smith's algorithm find a schedule that minimises \bar{C} subject to the constraint that no job can be late. Give the optimal value of \bar{C} and the corresponding schedule. **[8 marks]**

- (5) (a) An engineering firm has agreed to undertake the design, fabrication, and testing of a prototype transmission for a major car manufacturer. They have identified the following activities and their associated times and precedence relationships. Construct the network diagram that represents this project and find the critical path. **[7 marks]**

	ACTIVITY	TIME (WEEKS)	IMMEDIATE PREDECESSORS
A	Establish design specifications	1	-
B	Mechanical design	5	A
C	Electrical design	4	A
D	Final design review	1	B, C
E	Prepare test vehicle	2.5	B
F	Fabricate prototype	2	D
G	Conduct test	3.5	E, F
H	Prepare 'blueprints'	1	G
I	Prepare final report	1	H

With only one team of workers, the firm will take 21 weeks to finish the project. Show that by getting a second team in for some of the time to work on non-critical activities, the project can be finished within the total time of the critical path. Draw a Gantt diagram showing the feasibility of your plan. **[5 marks]**

- (b) For small extra cost, the time for activity B could be cut to 4 weeks and that of G to 3 weeks. What should the firm do if it could gain prestige and a handsome bonus by finishing the project within 13 weeks? Justify your answer. **[3 marks]**

(Questions done: 1, 2, 3)