CS711008Z Algorithm Design and Analysis

Lecture 9. Lagrangian duality and SVM

Dongbo Bu

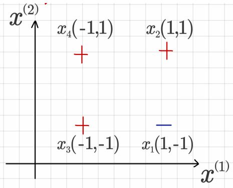
Institute of Computing Technology
Chinese Academy of Sciences, Beijing, China

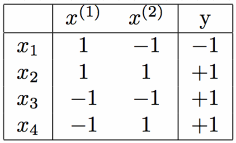
Outline

- Classification problem and maximum margin strategy;
- Solving maximum margin problem using Lagrangian duality;
- SMO technique;
- Kernel tricks;

Classification problem and maximum margin strategy

Classification problem

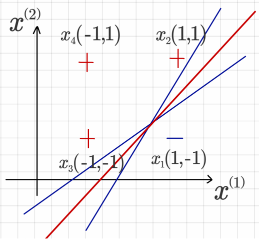


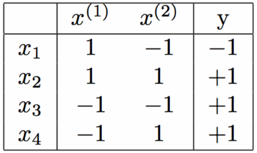


	$x^{(1)}$	$x^{(2)}$	y
x_1	1	-1	-1
x_2	1	1	$+1$
x_3	-1	-1	$+1$
x_4	-1	1	$+1$

- Given a set of samples with their category labels (denoted as $(\mathbf{x_1}, y_1), (\mathbf{x_2}, y_2), \ldots, (\mathbf{x_n}, y_n)$, $y_i \in \{-1, +1\}$, the goal of classification problem is to find an appropriate function $f(\mathbf{x})$ that can describe the dependency between y_i and $\mathbf{x_i}$; thus, for a new sample $\mathbf{x}'$, we can infer its category based on $f(\mathbf{x}')$.
- A great variety of classification algorithms have been designed, including Fisher's linear discriminant, logistic regression, decision tree, neural network and SVM.

Linear classifier



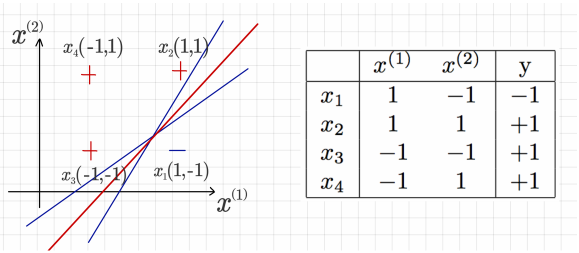


	$x^{(1)}$	$x^{(2)}$	y
x_1	1	-1	-1
x_2	1	1	$+1$
x_3	-1	-1	$+1$
x_4	-1	1	$+1$

- Unlike decision tree, SVM adopts the classifier with the following type:
  - If $f(\mathbf{x}) > 0$ then $y = +1$;
  - If $f(\mathbf{x}) < 0$ then $y = -1$;
- Let's first restrict the $f(\mathbf{x})$ to be linear, i.e.

$$f(\mathbf{x}) = \omega^T \mathbf{x} + b$$

The hyperplane $\omega^T \mathbf{x} + b = 0$ is denoted as separating hyperplane.



	$x^{(1)}$	$x^{(2)}$	y
x_1	1	-1	-1
x_2	1	1	$+1$
x_3	-1	-1	$+1$
x_4	-1	1	$+1$

- The objective of training procedure is to find an appropriate setting of ω and b such that all samples in the training set can be correctly labelled using the classifier. We will consider the torlerance of several mislabelled samples later.

Maximum margin strategy

- There are always multiple settings of ω and b that the corresponding classifier works perfectly on all samples. Which one should we use?
- We prefer the one such that the margin between positive and negative samples is maximized: The wider the margin is, the larger the generality performance on new samples. Thus, we needs to solve the following optimization problem:

$$\begin{array}{rl} \max_{w,b} & \frac{2}{||\omega||} \\ s.t. & y_i(w \cdot x_i + b) - 1 \geqslant 0 \quad i = 1, 2, \cdots, n \end{array}$$

- Note:
  - The restriction $f(\mathbf{x}) > 0$ for positive sample x is implemented as $f(\mathbf{x}) = 1$.
  - The distance for any point x to the hyperplane $\omega^T\mathbf{x} + b = 0$ is $\frac{|\omega^T\mathbf{x}+b|}{||\omega||}$. Thus, the margin is: $\frac{2}{||\omega||}$.

An equivalent form with quadratic objective function

- An equivalent form is:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\left\|w\right\|^2 \\ s.t. \quad & y_i(w \cdot x_i + b) - 1 \geqslant 0 \quad i = 1, 2, \cdots, n \end{aligned}$$

- Question: how to solve this optimization problem subject to inequality constraints?
- Of course we solve the problem (called primal problem hereafter) directly using convex quadratic programming techniques; however, consider its dual problem will bring great benefits.
- Let's review the conditions of the optimal solution first.

How to solve constrainted optimization problem?

Standard form of constrained optimization problems

- Consider the following constrained optimization problem (might be non-convex).

$$\begin{array}{llll} \min & f_0(x) & & \\ s.t. & f_i(x) & \leq & 0 \quad i = 1, ..., m \\ & h_i(x) & = & 0 \quad i = 1, ..., k \end{array}$$

- Here the variables $x \in \mathbb{R}^n$ and we use $\mathcal{D} = \bigcap_{i=0}^{m} \textbf{dom } f_i \cap \bigcap_{i=1}^{k} \textbf{dom } h_i$ to represent the domain of definition. We use p^* to represent the optimal value of the problem.

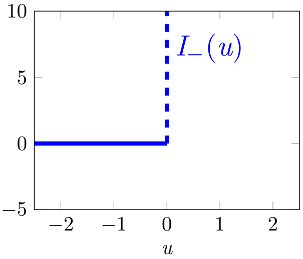
An equivalent unconstrained optimization problem

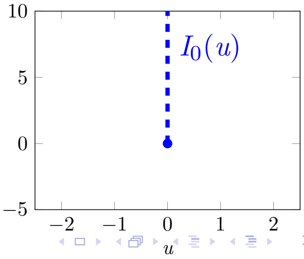
- We can transform this **constrained optimization** problem into an equivalent **unconstrained optimization** problem:

$$\min f_0(x) + \sum_{i=1}^{m} I_-(f_i(x)) + \sum_{i=1}^{k} I_0(h_i(x))$$

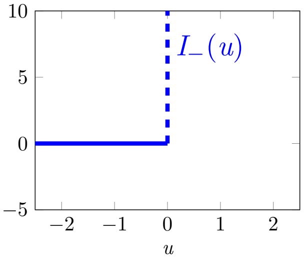
where $x \in \mathcal{D}$, $I_-(u)$ and $I_0(u)$ are indicator functions for non-positive reals and the set $\{0\}$, respectively:

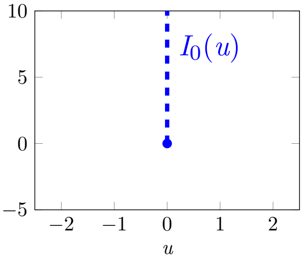
$$I_-(u) = \begin{cases} 0 & u \leq 0 \\ \infty & u > 0 \end{cases} \qquad I_0(u) = \begin{cases} 0 & u = 0 \\ \infty & u \neq 0 \end{cases}$$





Difficulty in solving the optimization problem



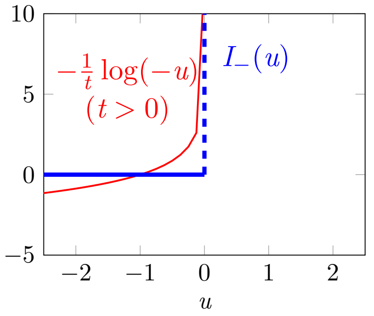


- Intuitively, $I_-(u)$ and $I_0(u)$ represent our "infinite dissatisfaction" with the violence of constraints.
- However both $I_0(u)$ and $I_-(u)$ are non-differentiable, making the optimization problem, although unconstrained, not easy to solve.

$$\min f_0(x) + \sum_{i=1}^{m} I_-(f_i(x)) + \sum_{i=1}^{k} I_0(h_i(x))$$

- Question: How to efficiently solve this optimization problem?

Approximating $I_-(u)$ using a differentiable function (1)

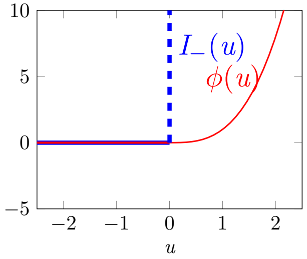


- An approximation to $I_-(u)$ is **logarithm barrier function**:

$$\hat{I}_-(u) = -\frac{1}{t}\log(-u) \qquad (t > 0)$$

- The difference between $\hat{I}_-(u)$ and $I_-(u)$ decreases as t increases. This approximation was used in the interior point method.

Approximating $I_-(u)$ using a differentiable function (2)

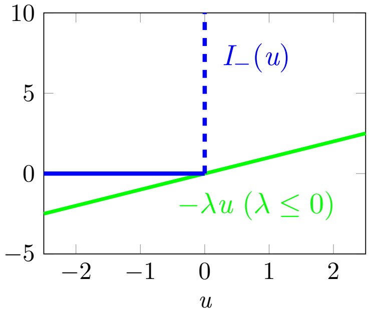


- Another approximation to $I_-(u)$ is a **penalty function**:

$$\hat{I}_-(u) = \phi(u) = \begin{cases} u^t & u \geq 0 \\ 0 & otherwise \end{cases} \qquad (t > 1)$$

- The penalty function "penalizes" any u if it is greater than zero. It is a "hands-off" method for converting constrainted problems into unconstrained problems, to which an initial feasible solution is easy to obtained. However, in some cases it cannot be applied because the objective function is undefined or the unconstrained problem becomes ill-conditioned as t increases.

Approximating $I_-(u)$ using a differentiable function (3)

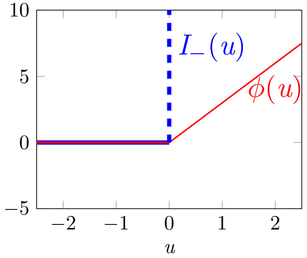


- Another approximation to $I_-(u)$ is a simple **linear function**:

$$\hat{I}_-(u) = -\lambda u \qquad (\lambda \leq 0)$$

- Despite the considerable difference between $\hat{I}_-(u)$ and $I_-(u)$, $\hat{I}_-(u)$ still provides lower bound information of $I_-(u)$.

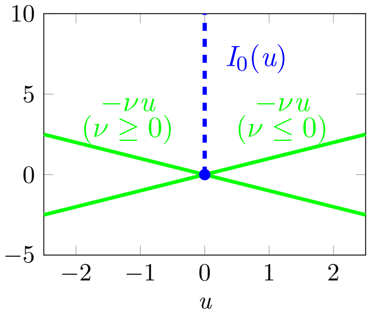
Approximating $I_{-}(u)$ using a differentiable function (4)



- We can also approximate $I_{-}(u)$ using **ReLU**:

$$\hat{I}_{-}(u) = \phi(u) = \begin{cases} ku & u \geq 0 \\ 0 & otherwise \end{cases} \qquad (k > 0)$$

Approximating $I_0(u)$ using a differentiable function



- $I_0(u)$ can also be approximated using **linear function**:

$$\hat{I}_0(u) = -\nu u$$

- Although $\hat{I}_0(u)$ deviates considerably from $I_0(u)$, $\hat{I}_0(u)$ still provides lower bound information of $I_0(u)$.
- It is worth pointing out that unlike $\hat{I}_-(u)$, $\hat{I}_0(u)$ has no restriction on ν.

Applying Lagrangian dual to the maximum margin problem

An example

- Primal problem:

$$\begin{array}{lccc} \min & x & & \\ s.t. & x & \geq & 2 \\ & x & \geq & 0 \end{array}$$

- Lagrangian:

$$L(x,y,z) = x - y*(x-2) - z*x = 2y + x*(1-y-z)$$

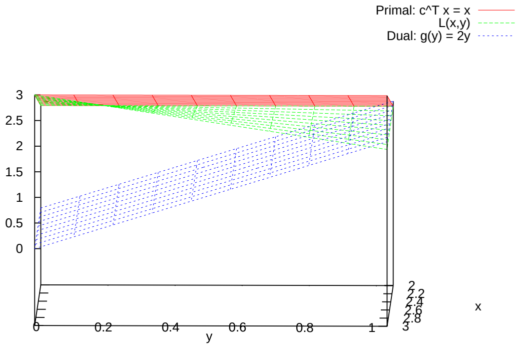
- Notice that when $y \geq 0$, $z \geq 0$ and $x \geq 2$, $L(x,y,z)$ is a lower bound of the primal objective function x.
- Lagrangian dual function:

$$g(y,z) = \inf_x L(x,y,z) = \begin{cases} 2y & \text{if } 1-y-z=0 \\ -\infty & \text{otherwise} \end{cases}$$

- Dual problem:

$$\begin{array}{lr} \max & 2y \\ s.t. & y \leq 1 \\ & y \geq 0 \end{array}$$

Lagrangian connecting primal and dual



- Observation: PRIMAL objective function $x \geq$ Lagrangian $\geq$ DUAL objective function y in the feasible region.

Lagrangian dual explanation of LP duality

- Consider a LP model:

$$\begin{array}{rrcl} \min & \mathbf{c}^{\mathbf{T}}\mathbf{x} & & \\ s.t. & \mathbf{A}\mathbf{x} & \geq & \mathbf{b} \\ & \mathbf{x} & \geq & \mathbf{0} \end{array}$$

- Lagrangian:

$$L(\mathbf{x}, \lambda, \mathbf{s}) = \mathbf{c}^{\mathbf{T}}\mathbf{x} - \sum_{i=1}^{m} \lambda_i(a_{i1}x_1 + \ldots + a_{in}x_n - b_i) - \sum_{i=1}^{m} s_i x_i$$

- Notice that **Lagrangian is a lower bound of the primal objective function**, i.e. $\mathbf{c}^{\mathbf{T}}\mathbf{x} \geq L(\mathbf{x}, \lambda, \mathbf{s})$, when $\lambda \geq \mathbf{0}$, $\mathbf{s} \geq \mathbf{0}$ and $\mathbf{x}$ is feasible.
- Furthermore we have

$$\mathbf{c}^{\mathbf{T}}\mathbf{x} \geq L(\mathbf{x}, \lambda) \geq \inf_{\mathbf{x}} L(\mathbf{x}, \lambda)$$

when $\lambda \geq \mathbf{0}$, $\mathbf{s} \geq \mathbf{0}$ and $\mathbf{x}$ is feasible.
- Denote **Lagrangian dual** $g(\lambda, \mathbf{s}) = \inf_{\mathbf{x}} L(\mathbf{x}, \lambda, \mathbf{s})$. The above inequality can be rewritten as:

$$\mathbf{c}^{\mathbf{T}}\mathbf{x} \geq L(\mathbf{x}, \lambda, \mathbf{s}) \geq g(\lambda, \mathbf{s})$$

Lagrangian dual explanation of LP duality cont'd

- What is the Lagrangian dual $g(\lambda, \mathbf{s})$?

$$\begin{aligned} g(\lambda, \mathbf{s}) &= \inf_{\mathbf{x}} L(\mathbf{x}, \lambda, \mathbf{s}) \\ &= \inf_{\mathbf{x}} (\mathbf{c}^{\mathbf{T}}\mathbf{x} - \sum_{i=1}^{m} \lambda_i (a_{i1}x_1 + \ldots + a_{in}x_n - b_i) - \sum_{i=1}^{m} s_i x_i) \\ &= \inf_{\mathbf{x}} (\lambda^{\mathbf{T}}\mathbf{b} + (\mathbf{c}^{\mathbf{T}} - \lambda^{\mathbf{T}}\mathbf{A} - \mathbf{s}^{\mathbf{T}})\mathbf{x}) \\ &= \begin{cases} \lambda^{\mathbf{T}}\mathbf{b} & \text{if } \mathbf{c}^{\mathbf{T}} = \lambda^{\mathbf{T}}\mathbf{A} + \mathbf{s}^{\mathbf{T}} \\ -\infty & \text{otherwise} \end{cases} \end{aligned}$$

- Thus $\lambda^{\mathbf{T}}\mathbf{b}$ is a lower bound of $f(\mathbf{x})$ when $\mathbf{c}^{\mathbf{T}} = \lambda^{\mathbf{T}}\mathbf{A} + \mathbf{s}^{\mathbf{T}}$ and $\mathbf{x} \geq \mathbf{0}$.
- Note $g(\lambda)$ is always concave even if the constraints are not convex.

Find a tight bound

- Now let's try to find the **best** lower bound of $f(\mathbf{x})$.
- Thus the tight lower bound $\max g(\lambda)$ can be described as:

$$\begin{array}{rrcl} \max & \lambda^{\mathbf{T}}\mathbf{b} & & \\ s.t. & \lambda^{\mathbf{T}}\mathbf{A} & \leq & \mathbf{c}^{\mathbf{T}} \\ & \lambda & \geq & \mathbf{0} \end{array}$$

- Notes:
  1. This is actually the DUAL form of LP if replacing λ by $\mathbf{y}$; thus, we have another explanation of DUAL variables $\mathbf{y}$ — the Lagrangian multiplier.
  2. Lagrangian dual is a lower bound of the primal optimum under some conditions. In addition, Lagrangian dual is concave; thus, **the dual problem is always a convex programming problem** even if the primal problem is not a convex programming problem.

Lagrangian dual explanation of maximum margin problem

- Primal problem:

$$\begin{array}{rrcl} \min_{w,b} & \frac{1}{2}\left\|w\right\|^2 & & \\ s.t. & y_i(w^T x_i + b) & \geq & 1, \quad i \in \{1, ..., n\} \end{array}$$

- Lagrangian:

$$L(w, b, \alpha) = \frac{1}{2}\left\|w\right\|^2 - \sum_{i=1}^{n} \alpha_i y_i (w \cdot x_i + b) + \sum_{i=1}^{n} \alpha_i$$

- Notice that **Lagrangian is a lower bound of the primal objective function**, i.e. $\frac{1}{2}\left\|w\right\|^2 \geq L(w, b, \alpha)$, when $\alpha \geq \mathbf{0}$ and $\mathbf{w}, \mathbf{b}$ is feasible.
- Furthermore we have

$$\frac{1}{2}\left\|w\right\|^2 \geq L(w, b, \alpha) \geq \inf_{w,b} L(w, b, \alpha)$$

  when $\alpha \geq 0$ and w, b is feasible.
- Denote **Lagrangian dual** $g(\alpha) = \inf_{w,b} L(w, b, \alpha)$. The above inequality can be rewritten as:

$$\frac{1}{2}\left\|w\right\|^2 \geq L(w, b, \alpha) \geq g(\alpha)$$

Lagrangian dual explanation of maximum margin duality cont'd

- What is the Lagrangian dual $g(\alpha)$?

$$g(\alpha) = \inf_{w,b} L(w, b, \alpha)$$

- To calculate the inferior bound of $L(w, b, \alpha)$, we set its derivates to be 0, i.e.,

$$\frac{\partial L(w, b, \alpha)}{\partial w} = w - \sum_{i=1}^{n} \alpha_i y_i x_i = 0$$

$$\frac{\partial L(w, b, \alpha)}{\partial b} = \sum_{i=1}^{n} \alpha_i y_i = 0$$

Lagrangian dual function

- Lagrangian dual function:

$$g(\alpha) = -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j (x_i^T x_j) + \sum_{i=1}^{n}\alpha_i$$

- Thus $g(\alpha)$ is a lower bound of $\frac{1}{2}\left\|w\right\|^2$ when $\sum_{i=1}^{n}\alpha_i y_i = 0$ and $\alpha \geq 0$.

Lagrangian dual problem

- Now let's try to find **the tightest lower bound** of $\frac{1}{2}\|w\|^2$, which can be calculated by solving the following Lagrangian dual problem:

$$\begin{array}{rrcl} \max & -\frac{1}{2}\sum_{i=1}^n \sum_{j=1}^n \alpha_i\alpha_j y_i y_j (x_i^T x_j) + \sum_{i=1}^n \alpha_i & & \\ s.t. & \sum_{i=1}^n \alpha_i y_i & = & 0 \\ & \alpha & \geq & 0 \end{array}$$

Two explanations of dual variables $\mathbf{y}$: L. Kantorovich vs. T. Koopmans

1. Price interpretation: constrained optimization plays an important role in economics. Dual variables are also called as **shadow price** (by T. Koopmans), i.e. the instantaneous change in the optimization objective function when constraints are relaxed, or **marginal cost** when strengthening constraints.
2. **Lagrangian multiplier**: the effect of constraints on the objective function (by L. Kantorovich). For example, when b_i increase to $b_i + \Delta b_i$, how much the objective function value will change. In fact, we have $\frac{\partial L(\mathbf{x},\lambda)}{\partial b_i} = \lambda_i$.

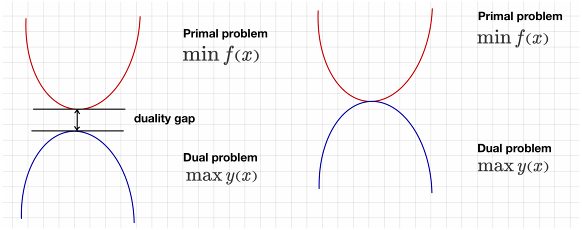
Explanation of dual variables y: using DIET as an example

- Optimal solution to primal problem with $b_1 = 2000, b_2 = 55, b_3 = 800$:
  $$\mathbf{x} = (14.24, 2.70, 0, 0),$$
  $$\mathbf{c}^\mathbf{T}\mathbf{x} = 67.096.$$
- Optimal solution to dual problem:
  $$\mathbf{y} = (0.0269, 0, 0.0164),$$
  $$\mathbf{y}^\mathbf{T}\mathbf{b} = 67.096.$$
- Let's make a slight change on $\mathbf{b}$, and watch the effect on $\max \mathbf{c}^\mathbf{T}\mathbf{x}$.
  1. $b_1 = 2001$: $\max \mathbf{c}^\mathbf{T}\mathbf{x} = 67.123$ (Note that $\mathbf{y_1} = 0.0269 = 67.123 - 67.096$)
  2. $b_2 = \quad 56$: $\max \mathbf{c}^\mathbf{T}\mathbf{x} = 67.096$ (Note that $\mathbf{y_2} = 0 = 67.096 - 67.096$)
  3. $b_3 = \quad 801$: $\max \mathbf{c}^\mathbf{T}\mathbf{x} = 67.112$ (Note that $\mathbf{y_3} = 0.0164 = 67.112 - 67.096$)

(See extra slides)

Weak duality, strong duality, and Slater conditions

Weak duality and strong duality



- Sometimes the optimal solution of the primal problem doesn't has identical value to that of the dual problem. The difference between them is denoted as duality gap. This case is denoted as weak duality. The opposite case is denoted as strong duality.
- Conditions under which strong duality holds are denoted as **constraint qualifications**.

Slater's conditions for strong duality

- The best known sufficient condition of strong duality is:
  - Primal problem is convex programming, i.e.,

$$\begin{array}{llll} \min & f(\mathbf{x}) & & \\ s.t. & g_i(\mathbf{x}) & \leq & 0 \quad i = 1, 2, ...m \\ & \mathbf{Ax} & = & \mathbf{b} \end{array}$$

    where $f(x)$ and $g_i(x)$ are convex.
  - Slater's condition holds, i,e. there exists some strictly feasible point $x \in relint(\mathcal{D})$ such that

$$\mathbf{Ax} = \mathbf{b}$$

    and

$$g_i(\mathbf{b}) < 0, i = 1, 2, ..., m$$

    Here $\mathcal{D}$ represents the feasible domain of the primal problem, and $relint(\mathcal{D})$ denotes the relative interior of $\mathcal{D}$.

Slater's conditions for strong duality (cont'd)

- The Slater's conditions are trivial if the constraints of the primal problem are affine, i.e., Consider the following convex programming problem:

$$\begin{array}{rrcll} \min & f(\mathbf{x}) & & & \\ s.t. & g_i(\mathbf{x}) & \leq & 0 & i = 1, 2, ...m \\ & \mathbf{Ax} & = & \mathbf{b} & \end{array}$$

  where $f(x)$ and $g_i(x)$ are convex.

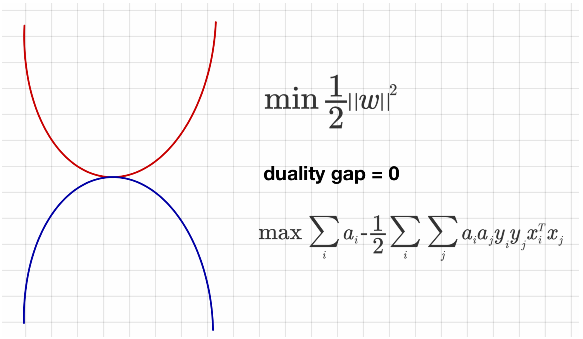
- Slater's condition: If $g_i(x)$ are affine, the inequalities are no longer required to be strict and thus reduce to the original form, i.e. there exists some strictly feasible point $x \in relint(\mathcal{D})$ such that

$$\mathbf{Ax} = \mathbf{b}$$

  and

$$g_i(\mathbf{b}) \leq 0, i = 1, 2, ..., m$$

Strong duality in SVM

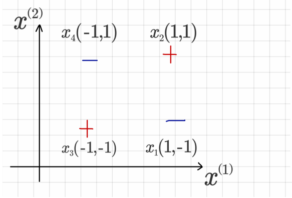


$\min \frac{1}{2}||w||^2$
duality gap = 0
$\max \sum_i a_i - \frac{1}{2}\sum_i \sum_j a_i a_j y_i y_j x_i^T x_j$

- For the maximum margin problem, its dual problem has the identical optimal objective function value.

Kernel method to overcome linear insperability

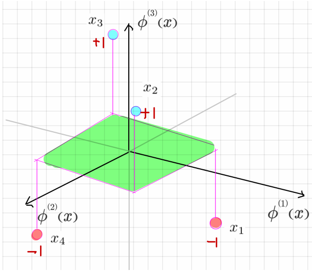
Linearly inseperable



	$x^{(1)}$	$x^{(2)}$	y
x_1	1	-1	-1
x_2	1	1	$+1$
x_3	-1	-1	$+1$
x_4	-1	1	-1

- Some samples are linearly insuperable. How should we do?
- Kernel method: If the samples are linearly inseperable in the original input space, let's map them into a higher-dimensional space (called feature space) and make them linearly separable in that space. In other words, nonlinearity was introduced into the original input space through using the map.

An example: XOR problem



	$\phi^{(1)}(x)$	$\phi^{(2)}(x)$	$\phi^{(3)}(x)$	y
x_1	1	-1	-1	-1
x_2	1	1	1	$+1$
x_3	-1	-1	1	$+1$
x_4	-1	1	-1	-1

- Let's define a map: $\phi : \mathbb{R}^2 \to \mathbb{R}^3$ as below:

$$[x^{(1)}, x^{(2)}] \to \Phi([x^{(1)}, x^{(2)}]) = [x^{(1)}, x^{(2)}, x^{(1)}x^{(2)}]$$

- These samples are linearly seperable after mapping into $\mathbb{R}^3$ using $\Phi(x^{(1)}, x^{(2)})$

Replace the inner product in the optimization problem for SVM

- Remember that in the optimization problem for SVM, the objective function contains inner product $\mathbf{x}^T\mathbf{x}$:

$$\begin{array}{ll} \max & -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(x_i^T x_j) + \sum_{i=1}^{N}\alpha_i \\ s.t. & \sum_{i=1}^{N}\alpha_i y_i = 0 \\ & \alpha \geq \mathbf{0} \end{array}$$

- After mapping $\mathbf{x}$ into $(\mathbf{x})$, the inner product $\mathbf{x}^T\mathbf{x}$ changes into $(\mathbf{x})^T(\mathbf{x})$, and the optimization problem changes accordingly:

$$\begin{array}{ll} \max & -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(\Phi(x_i)^T\Phi(x_j)) + \sum_{i=1}^{N}\alpha_i \\ s.t. & \sum_{i=1}^{N}\alpha_i y_i = 0 \\ & \alpha \geq \mathbf{0} \end{array}$$

  Note we simply replace this inner product with $(\mathbf{x})^T(\mathbf{x})$ **without even knowing the map $(\mathbf{x})$**.

- In other words, we define a kernel function

$$k(\mathbf{x}_i, \mathbf{x}_j) = < (\mathbf{x}_i), (\mathbf{x}_j) >_{\mathcal{H}}$$

The seperating hyperplane

- Remember that in the optimization problem for SVM, the seperating hyperplane is:

$$f(\mathbf{x}) = \omega^T \mathbf{x} + b = \sum_{i,j} \alpha_i y_i \mathbf{x}_i^T \mathbf{x} = 0$$

- After mapping $\mathbf{x}$ into $\ (\mathbf{x})$, the sporting hyperplane changes into:

$$f(\mathbf{x}) = \omega^T \ (\mathbf{x}) + b = \sum_{i,j} \alpha_i y_i \ (\mathbf{x})_i^T \ (\mathbf{x}) = 0$$

- Note that $\ (\mathbf{x})_i^T \ (\mathbf{x}) = k(\ (\mathbf{x})_i, \ (\mathbf{x}))$.

Deriving kernel function from map

Example 1

- Consider the map $\Phi : \mathbb{R}^2 \rightarrow \mathbb{R}^4$:

$$[x^{(1)}, x^{(2)}] \rightarrow \Phi([x^{(1)}, x^{(2)}]) = [x^{(1)2}, x^{(2)2}, x^{(1)}x^{(2)}, x^{(1)}x^{(2)}]$$

- The corresponding kernel function is:

$$k(\mathbf{x}, \mathbf{z}) = x^{(1)2}z^{(1)2} + x^{(2)2}z^{(2)2} + 2x^{(1)}x^{(2)}z^{(1)}z^{(2)} =< \mathbf{x}, \mathbf{z} >^2_{\mathbb{R}^2}$$

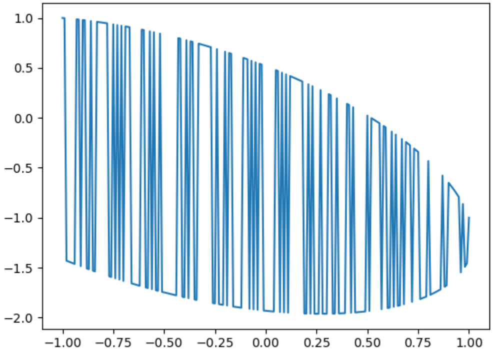
Example 2

- Consider the map $\Phi : \mathbb{R}^3 \rightarrow \mathbb{R}^{13}$:

  $[x^{(1)}, x^{(2)}, x^{(3)}] \rightarrow \Phi([x^{(1)}, x^{(2)}, x^{(3)}]) = [x^{(1)2}, x^{(1)}x^{(2)}, ..., x^{(3)2}, \sqrt{2c}x^{($

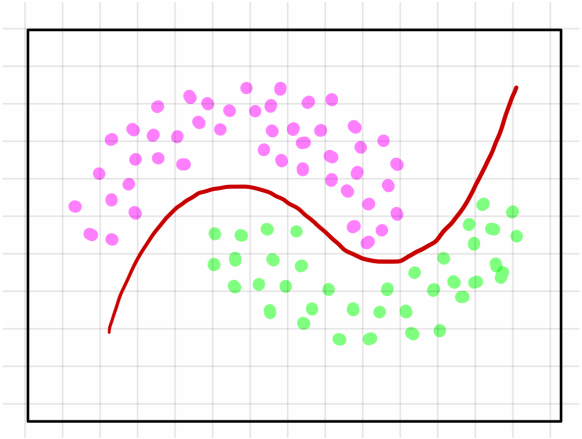
- The corresponding kernel function is:

$$k(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T\mathbf{z} + c$$



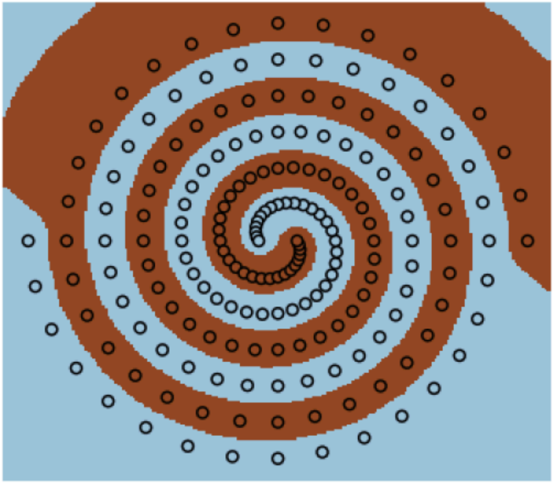
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

The power of polynomial kernel



- We can map into a d-dimensional feature space.

The power of polynomial kernel



The requirement of kernel funcitons

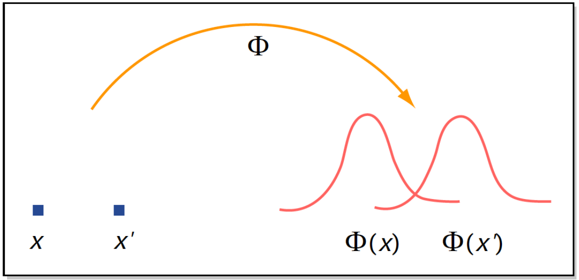
Definition of kernel function

- A function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is a kernel function if:
  - k is symmetric: $k(x, z) = k(z, x)$.
  - For any $m \in N$ and any $x_1, x_2, ..., x_m \in mathcalX$, the matrix K defined by $K_{i,j} = k(x_i, x_j)$ is positive semidefinite.

$$
\mathrm{K}= \begin{array}{c|ccc}
 & 1 & j & m \\
\hline
m & & & \\
i & & k(x_i,x_j) & \\
1 & & &
\end{array}
$$

Question: Could we map from input space into a feature space with infinite dimension?

Map into an infinite dimensional feature space



- Let's consider the feature space where all elements are functions.
- The span of this space is $\Phi(x_i) = k(., x_i)$. Thus, any element can be represented as:

$$f(\cdot) = \sum_{i=1}^{m} \alpha_i k(\cdot, x_i) \leftarrow \text{“vectors”}$$

- For two elements

$$\langle f, g \rangle_{H_k} = \sum_{i=1}^{m} \sum_{j=1}^{m'} \alpha_i \beta_j k(x_i, x_j').$$

we define inner product as:

$$f(\cdot) = \textstyle\sum_{i=1}^{m} \alpha_i k(\cdot, x_i) \text{ and } g(\cdot) = \sum_{j=1}^{m'} \beta_j k(\cdot, x_j')$$

Reproducing kernel Hilbert space

- Now we find a map such that

$$\langle k(\cdot, x), k(\cdot, x') \rangle_{H_k} = k(x, x')$$

Theorem (Farkas lemma)

Given vectors $\mathbf{a_1}, \mathbf{a_2}, ..., \mathbf{a_m}, \mathbf{c} \in \mathbb{R}^n$. *Then either*

1. $\mathbf{c} \in \mathbf{C}(\mathbf{a_1}, \mathbf{a_2}, ..., \mathbf{a_m})$*; or*
2. *there is a vector* $\mathbf{y} \in \mathbb{R}^n$ *such that for all* i, $\mathbf{y^T a_i} \geq \mathbf{0}$ *but* $\mathbf{y^T c} < \mathbf{0}$.

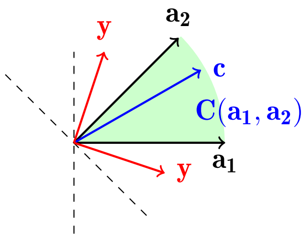


Figure: Case 1: $\mathbf{c} \in \mathbf{C}(\mathbf{a_1}, \mathbf{a_2})$

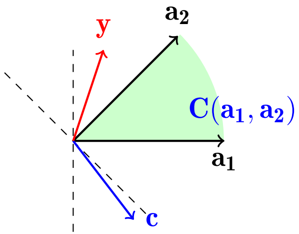


Figure: Case 2: $\mathbf{c} \notin \mathbf{C}(\mathbf{a_1}, \mathbf{a_2})$

- Here, $\mathbf{C}(\mathbf{a_1}, ..., \mathbf{a_m})$ denotes the cone spanned by $\mathbf{a_1}, ..., \mathbf{a_m}$, i.e. $\mathbf{C}(\mathbf{a_1}, ..., \mathbf{a_m}) = \{\mathbf{x} | \mathbf{x} = \sum_{i=1}^{m} \lambda_i \mathbf{a_i}, \lambda_i \geq 0\}$.

Proof.

- Suppose for any vector $\mathbf{y} \in \mathbb{R}^n$, $\mathbf{y^T a_i} \geq \mathbf{0}$ $(i = 1, 2, ..., m)$, we always have $\mathbf{y^T c} \geq \mathbf{0}$. We will show that $\mathbf{c}$ should lie within the cone $\mathbf{C}(\mathbf{a_1}, \mathbf{a_2}, ..., \mathbf{a_m})$.
- Consider the following PRIMAL problem:

$$\begin{array}{lllll} \min & \mathbf{c^T y} & & & \\ s.t. & \mathbf{a_i^T y} & \geq & \mathbf{0} & i = 1, 2, ..., m \end{array}$$

- It is obvious that the PRIMAL problem has a feasible solution $\mathbf{y} = \mathbf{0}$, and is bounded since $\mathbf{c^T y} \geq \mathbf{0}$.
- Thus the DUAL problem also has a bounded optimal solution:

$$\begin{array}{lrcl} \max & 0 & & \\ s.t. & \mathbf{x^T A^T} & = & \mathbf{c^T} \\ & \mathbf{x} & \geq & \mathbf{0} \end{array}$$

- In other words, there exists a vector $\mathbf{x}$ such that $\mathbf{c} = \sum_{i=1}^{m} x_i \mathbf{a_i}$ and $x_i \geq 0$.

Variants of Farkas' lemma

Farkas' lemma lies at the core of linear optimization. Using Farkas' lemma, we can prove SEPARATION theorem, and MINIMAX theorem in the game theory.

Theorem

Let $\mathbf{A}$ *be an* $m \times n$ *matrix, and* $\mathbf{b} \in \mathbb{R}^m$*. Then either*

1. $\mathbf{A}\mathbf{x} = \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$ *has a feasible solution; or*
2. *there is a vector* $\mathbf{y} \in \mathbb{R}^m$ *such that* $\mathbf{y}^\mathbf{T}\mathbf{A} \geq \mathbf{0}$ *but* $\mathbf{y}^\mathbf{T}\mathbf{b} < \mathbf{0}$*.*

Variants of Farkas' lemma

Theorem

Let $\mathbf{A}$ *be an* $m \times n$ *matrix, and* $\mathbf{b} \in \mathbb{R}^m$. *Then either*

1. $\mathbf{A}\mathbf{x} \leq \mathbf{b}$ *has a feasible solution; or*
2. *there is a vector* $\mathbf{y} \in \mathbb{R}^m$ *such that* $\mathbf{y} \geq \mathbf{0}$, $\mathbf{y}^{\mathbf{T}}\mathbf{A} \geq \mathbf{0}$ *but* $\mathbf{y}^{\mathbf{T}}\mathbf{b} < \mathbf{0}$.

Caratheodory's theorem

Theorem

Given vectors $\mathbf{a_1}, \mathbf{a_2}, ..., \mathbf{a_m} \in \mathbb{R}^n$*. If* $\mathbf{x} \in \mathbf{C}(\mathbf{a_1}, \mathbf{a_2}, ..., \mathbf{a_m})$*, then there is a linearly independent vector set of* $\mathbf{a_1}, \mathbf{a_2}, ..., \mathbf{a_m}$*, say* $\mathbf{a_1}, \mathbf{a_2}, ..., \mathbf{a_r}$*, such that* $\mathbf{x} \in \mathbf{C}(\mathbf{a_1}, \mathbf{a_2}, ..., \mathbf{a_r})$*.*

Separation theorem

Theorem

Let $\mathbf{C} \subseteq \mathbb{R}^n$ *be a closed, convex set, and let* $\mathbf{x} \in \mathbb{R}^n$*. If* $\mathbf{x} \notin \mathbf{C}$*, then there exists a hyperplane separating* $\mathbf{x}$ *from* $\mathbf{C}$*.*

Application 2: von Neumann's MiniMax theorem on game theory

Game theory

- Game theory studies competing and cooperative behaviours among intelligent and rational decision-makers.
- In 1928, John von Neumann proved the existence of mixed-strategy equilibria in two-person zero-sum games.
- In 1950, John Forbes Nash Jr. developed a criterion of mutual consistency of players' strategies, which applies to a wider range of games than that proposed by J. von Neumann. He proved the existence of Nash equilibrium in every n-player, non-zero-sum, non-cooperative game (not just 2-player, zero-sum games).
- Game theory was widely applied in mathematical economics, in biology (e.g., analysis of evolution and stability) and computer science (e.g., analysis of interactive computations and lower bound on the complexity of randomized algorithms, the equivalence between linear program and two-person zero-sum game).

Paper-rock-scissors: an example of two-player zero-sum game

- Paper-rock-scissors is a hand game usually played by two players, denoted as row player and column player: each player selects one of the three hand shapes, including “paper”, “rock”, and “scissors”; then the players show their selections simultaneously.
- It has two possible outcomes other than tie: one player wins and the other player loses, which can be formally described using the following payoff matrix.

	Paper	Rock	Scissors
Paper	0, 0	1, -1	-1, 1
Rock	-1, 1	0, 0	1, -1
Scissors	1, -1	-1, 1	0, 0

- Each player attempts to select appropriate action to maximize his gain.

Matching penny: another example of two-person zero-sum game

- Matching pennies is a game played by two players, namely, row player and column player. Each player has a penny and secretly turns it to head or tail. The players then reveal their selections simultaneously.
- If the pennies match, then row player keeps both pennies; otherwise, column player keeps both. The payoff matrix is as follows.

	Head	Tail
Head	1, -1	-1, 1
Tail	-1, 1	1, -1

- Each player tries to maximize his gain via making an appropriate selection.

Simultaneous games vs. sequential games

- **Simultaneous games** are games in which all players move **simultaneously**. Thus, no player have information of the others' selections in advance.
- **Sequential games** are games in which the later player has some information, although maybe imperfect, of previous actions by the other players. A complete plan of action for every stage of the game, regardless of whether the action actually arises in play, is denoted as **a (pure) strategy**.
- **Normal form** is used to describe simultaneous games while **extensive form** is used to describe sequential games.
- J. von Neumann proposed an approach to transform transform strategies in sequential games into actions in simultaneous games.
- Note that the transformation is one-way, i.e., multiple sequential games might correspond to the same simultaneous game, and it may result in an exponential blowup in the size of the representation.

Normal form

- A game Γ in normal form among m players contains the following items:
  - Each player k has a finite number of **pure strategies** $S_k = \{1, 2, ..., n_k\}$.
  - Each player k is associated with a **payoff function** $H_k : S_1 \times S_2 \times ... \times S_m \to \mathbb{R}$.
- To play the game, each player selects a strategy without information of others, and then reveals the selection **simultaneously**. The players' gain are calculated using corresponding payoff functions.
- Each player attempts to maximize his gain via selecting an appropriate strategy.

Two-person zero-sum game in normal form

- In a two-person zero-sum game game Γ, a player's gain or less is exactly balanced by the other player's loss or gain, i.e.,

$$H_1(s_1, s_2) + H_2(s_1, s_2) = 0.$$

- Thus we can define another function

$$H(s_1, s_2) = H_1(s_1, s_2) = -H_2(s_1, s_2)$$

and represent it using a **payoff matrix**.

	Head	Tail
Head	1	-1
Tail	-1	1

- Row player aims to maximize $H(s_1, s_2)$ by selecting an appropriate strategy s_1 while column player aims to minimize $H(s_1, s_2)$ by selecting an appropriate strategy s_2.

von Neumann's MINIMAX theorem: motivation

- When analyzing a two-person zero-sum game Γ, von Neumann noticed that the difficulty comes from the difference between games and ordinary optimization problems: row player tries to maximize $H(s_1, s_2)$; however, he can control s_1 only as he has no information of the other player's selection s_2, and so does column player.
- Thus von Neumann suggested to investigate two auxiliary games without this difficulty, denoted as Γ_1 and Γ_2, before attacking the challenging game Γ.
    1. Γ_1: Row player selects a strategy s_1 first, and exposes his selection to column player before column player selects a strategy s_2.
    2. Γ_2: Column player selects a strategy s_2 first, and exposes his selection to row player before row player selects a strategy s_1.
- The two auxiliary games are much easier than the original game Γ, and more importantly, they provide upper and lower bounds for Γ.

Auxiliary game Γ_1

- Let's consider column player first. As he knows row player's selection s_1, the objective function $H(s_1, s_2)$ becomes an ordinary optimization function over a single variable s_2, and column player can simply select a strategy s_2 with the minimum objective function value $\min_{s_2} H(s_1, s_2)$.

	Head	Tail	Row minimum
Head	-2	1	-2
Tail	-1	2	$v_1 = -1$

- Now consider row player. When he selects a strategy s_1, he can definitely predict the selection of column player. Since $\min_{s_2} H(s_1, s_2)$ is an ordinary function over a single s_1, it is easy for row player to select a strategy s_1 with the maximum objective function value

$$v_1 = \max_{s_1} \min_{s_2} H(s_1, s_2).$$

Auxiliary game Γ_2

- Let's consider row player first. As he knows column player's selection s_2, the objective function $H(s_1, s_2)$ becomes an ordinary optimization function over a single variable s_1, and row player can simply select a strategy s_1 with the maximum objective function value $\max_{s_1} H(s_1, s_2)$.

	Head	Tail
Head	-2	1
Tail	-1	2
Column maximum	$v_2 = -1$	2

- Now consider column player. When he selects a strategy s_2, he can definitely predict the selection of row player. Since $\max_{s_1} H(s_1, s_2)$ is an ordinary function over a single variable s_2, it is easy for column player to select a strategy s_2 with the minimum objective function value

$$v_2 = \min_{s_2} \max_{s_1} H(s_1, s_2).$$

Γ_1 and Γ_2 bound Γ

- For row player, it is clearly Γ_1 is disadvantageous to him as he should expose his selection s_1 to column player.
- On the contrary, Γ_2 is beneficial to row player as he knows column player's selection s_2 before making decision.

	Head	Tail	Row minimum
Head	-2	1	-2
Tail	-1	2	$v_1 = -1$
Column maximum	$v_2 = -1$	2	

- Thus these two auxiliary games provides lower and upper bounds:

$$v_1 \le v \le v_2$$

where v denote row player's gain in the original game Γ.

Case 1: $v_1 = v_2$

- For a game with the following payoff matrix, we have $v_1 = v = v_2$ and call this game **strictly determined**.

	Head	Tail	Row minimum
Head	-2	1	-2
Tail	-1	2	$v_1 = -1$
Column maximum	$v_2 = -1$	2	

- The saddle point of the payoff matrix $H(s_1, s_2)$ represents a **pure strategy equilibrium**. In this equilibrium, each player has nothing to gain by changing only his own strategy. In addition, knowing the opponent's selection will bring no gain.
- von Neumann proved the existence of the optimal strategy in a perfect information two-person zero-sum game, e.g., chess. L. S. Shapley further showed that a finite two-person zero-sum game has a pure strategy equilibrium if every 2×2 submatrix of the game has a pure strategy equilibrium [**?**].

Case 2: $v_1 < v_2$

- In contrast, matching penny does not have a pure strategy equilibrium as there is no saddle point in the payoff matrix. So does the paper-rock-scissors game.

	Head	Tail	Row minimum
Head	1	-1	-1
Tail	-1	1	$v_1 = -1$
Column maximum	$v_2 = 1$	1	

- This fact implies that knowing the opponent's selection might bring gain; however, it is impossible to know the opponent's selection as the players reveal their selections simultaneously. In this case, let's play a mixed strategy rather than a pure strategy.

From pure strategy to mixed strategy

- A **mixed strategy** is an assignment of probability to pure strategies, allowing a player to **randomly select a pure strategy**.
- Consider the payoff matrix as below. If the row player select strategy A with probability 1, he is said to play a pure strategy. If he tosses a coin and select strategy A if the coin lands head and B otherwise, then he is said to play a mixed strategy.

	A	B
A	1	-1
B	-1	1

Two types of interpretation of mixed strategy

- From a player's viewpoint: J. von Neumann described the motivation underlying the introduction of mixed strategy as follows: since it is impossible to exactly know opponent's selection, a player could switch to **protect himself by "randomly selecting his own strategy"**, making it difficult for the opponent to know the player's selection. However, this interpretation came under heavy fire for lacking of behaviour supports: Seldom do people make choices following a lottery.
- From opponent's viewpoint: Robert Aumann and Adam Brandenburger interpreted mixed strategy of a player as **opponent's "belief" of the player's selection**. Thus, Nash equilibrium is an equilibrium of "belief" rather than actions.

Existence of mixed strategy equilibrium

- Consider a mixed strategy game: row player has m strategies available and he selects a strategy s_1 according to a distribution $\mathbf{u}$, while column player has n strategies available and he selects a strategy s_2 according to a distribution $\mathbf{v}$, i.e.,

$$\Pr(s_1 = i) = u_i, i = 1, ..., n \quad \Pr(s_2 = j) = v_j, j = 1, ..., m$$

  Here, $\mathbf{u}$ and $\mathbf{v}$ are independent.
- Thus the expected gain of row player is:

$$\sum_{i=1}^{m} \sum_{j=1}^{n} u_i H_{ij} v_j = \mathbf{u}^{\mathbf{T}} \mathbf{H} \mathbf{v}$$

- row player attempts to minimize $\mathbf{u}^{\mathbf{T}} \mathbf{H} \mathbf{v}$ via selecting an appropriate $\mathbf{u}$, while column player attempts to maximize it via selecting an appropriate $\mathbf{v}$.
- Now let's consider the two auxiliary games Γ_1 and Γ_2 again and answer the following questions: what happens if row player exposes his mixed strategy to column player? And if we reverse the order of the players?

von Neumann's MiniMax theorem [1928]

- This question has been answered by the von Neumann's MiniMax theorem.

Theorem

$$\max_{\mathbf{u}} \min_{\mathbf{v}} \mathbf{u}^{\mathbf{T}}\mathbf{H}\mathbf{v} = \min_{\mathbf{v}} \max_{\mathbf{u}} \mathbf{u}^{\mathbf{T}}\mathbf{H}\mathbf{v}$$

- The theorem states that knowing the other player's strategy will bring no gain in a mixed-strategy zero-sum game, and the order doesn't change the value.
- A **mixed-strategy Nash equilibrium** exists for **any two-person zero-sum game** with a finite set of actions. A Nash equilibrium in a two-player game is a pair of strategies, each of which is a best response to the other; i.e., each gives the player using it the highest possible payoff, given the other players' strategy.

von Neumann's MiniMax theorem: proof

- Let's consider the auxiliary game Γ_1 first, in which the strategy of row player, i.e., $\mathbf{u}$, was exposed to column player. This is of course beneficial to column player since he can select the optimal strategy $\mathbf{v}$ to minimize $\mathbf{u}^{\mathbf{T}}\mathbf{H}\mathbf{v}$, which is

$$\inf\{\mathbf{u}^{\mathbf{T}}\mathbf{H}\mathbf{v} | \mathbf{v} \geq \mathbf{0}, \mathbf{1}^{\mathbf{T}}\mathbf{v} = 1\} = \min_{j=1,\ldots,n} (\mathbf{u}^{\mathbf{T}}\mathbf{H})_j$$

- Thus row player should select $\mathbf{u}$ to maximize the above value, which can be formulated as a linear program:

$$\begin{array}{rlcl} \max & \min_{j=1,\ldots,n} (\mathbf{u}^{\mathbf{T}}\mathbf{H})_j & & \\ s.t. & \mathbf{1}^{\mathbf{T}}\mathbf{u} & = & 1 \\ & \mathbf{u} & \geq & \mathbf{0} \end{array}$$

von Neumann's MiniMax theorem: proof

- The linear program can be rewritten as below.

$$\begin{array}{lrcl} \max & s & & \\ s.t. & \mathbf{u}^{\mathbf{T}}\mathbf{H} & \geq & s\mathbf{1}^{\mathbf{T}} \\ & \mathbf{1}^{\mathbf{T}}\mathbf{u} & = & 1 \\ & \mathbf{u} & \geq & \mathbf{0} \end{array}$$

- Similarly we consider the auxiliary game Γ_2 and calculate the optimal strategy $\mathbf{v}$ by solving the following linear program.

$$\begin{array}{lrcl} \min & t & & \\ s.t. & \mathbf{H}\mathbf{v} & \leq & t\mathbf{1} \\ & \mathbf{1}^{\mathbf{T}}\mathbf{v} & = & 1 \\ & \mathbf{v} & \geq & \mathbf{0} \end{array}$$

- These two linear programs are both feasible and form Lagrangian dual. Thus they have the same optimal objective value according to the strong duality property.

An example: paper-rock-scissors game

- For the paper-rock-scissors game, we have the following two linear programs.
  - Linear program for Γ_1:

$$\begin{array}{rrcrcrl} \max & s & & & & & \\ s.t. & 0u_1 & - & u_2 & + & u_3 & \geq s \\ & u_1 & + & 0u_2 & - & u_3 & \geq s \\ & -u_1 & + & u_2 & + & 0u_3 & \geq s \\ & u_1 & + & u_2 & + & u_3 & = 1 \\ & u_1, & & u_2, & & u_3 & \geq 0 \end{array}$$

  - Linear program for Γ_2:

$$\begin{array}{rrcrcrl} \min & t & & & & & \\ s.t. & 0v_1 & + & v_2 & - & v_3 & \leq t \\ & -v_1 & + & 0v_2 & + & v_3 & \leq t \\ & v_1 & - & v_2 & + & 0v_3 & \leq t \\ & v_1 & + & v_2 & + & v_3 & = 1 \\ & v_1, & & v_2, & & v_3 & \geq 0 \end{array}$$

- The mixed strategy equilibrium is $\mathbf{u}^{\mathbf{T}} = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$ and $\mathbf{u}^{\mathbf{T}} = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$ with the game value 0.

Comments on the mixed strategy equilibrium by von Neumann

- Note that a mixed strategy equilibrium always exists no matter whether the payoff matrix $\mathbf{H}$ has a saddle point or not.
- Regardless of column player's selection, row player can select an appropriate strategy to guarantee his gain $v_1 \geq 0$.
- Regardless of row player's selection, column player can select an appropriate strategy to guarantee row player's gain $v_1 \leq 0$.
- Using the strategy $\mathbf{u}^{\mathbf{T}} = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$, row player can guarantee that he "won't lose", i.e., the probability of losing is less than the probability of winning.
- The strategy $\mathbf{u}^{\mathbf{T}} = [\frac{1}{3}, \frac{1}{3}, \frac{1}{3}]$ is designed for "protecting himself" rather than "attacking his opponent", i.e., it cannot be used to benefit from opponent's fault.

Application 3: Yao's MiniMax principle [1977]

Yao's MINIMAX principle

- Consider a problem Π. Let $\mathcal{A} = \{A_1, A_2, ..., A_n\}$ be algorithms to Π, and $\mathcal{I} = \{I_1, I_2, ..., I_m\}$ be the inputs with a given size. Let $T(A_i, I_j)$ be the running time of algorithm A_i on the input I_j.

		Algorithms	
		A_1	A_2
Inputs	I_1	T_{11}	T_{12}
	I_2	T_{21}	T_{22}

- Thus $\max\limits_{I_j \in \mathcal{I}} T(A_i, I_j)$ represents the worst-case time for the deterministic algorithm A_i.
- For a randomized algorithms, however, it is usually difficult to bound its expected running time on worst-case inputs.
- Yao's MINIMAX principle provides a technique to build lower bound for **the expected running time of any randomized algorithm on its worst-case input**.

Expected running time of a randomized algorithm A_q

- A "Las Vegas" randomized algorithm can be viewed as a distribution over all deterministic algorithms $\mathcal{A} = \{A_1, A_2, ..., A_n\}$.
- Specifically, let q be a distribution over $\mathcal{A}$, and A_q be a randomized algorithm chosen according to q, i.e., A_q refers to a deterministic algorithm A_i with probability q_i.
- Given a input I_j, the expected running time of A_q can be written as

$$E[T(A_q, I_j)] = \sum_{i=1}^{n} q_i T(A_i, I_j)$$

- Thus $\max_{I_j \in \mathcal{I}} E[T(A_q, I_j)]$ represents the expected running time of A_q on its worst-case input.

Expected running time of a deterministic algorithm A_i on random input

- Now consider a deterministic algorithm A_i running on random input.
- Let p be a distribution over $\mathcal{I}$, and I_p be a random input chosen from $\mathcal{I}$, i.e., I_p refers to I_j with probability p_j.
- Given a deterministic algorithm A_i, its expected running time on random input I_p can be written as

$$E[T(A_i, I_p)] = \sum_{j=1}^{m} p_j T(A_i, I_j)$$

- Thus $\min\limits_{A_i \in \mathcal{A}} E[T(A_i, I_p)]$ represents the expected running time of the best deterministic algorithm on the random input I_p.

Yao's MiniMax principle

Theorem

For any random input I_p and randomized algorithm A_q,

$$\min_{A_i \in \mathcal{A}} E[T(A_i, I_p)] \leq \max_{I_j \in \mathcal{I}} E[T(A_q, I_j)]$$

- To establish a lower bound for the expected running time of a randomized algorithm on its worst-case input, it suffices to find an appropriate distribution over inputs and prove that on this random input, no deterministic algorithm can do better than the randomized one.
- The power of this technique lies at the fact that one can choose any distribution over inputs and the lower bound is constructed based on deterministic algorithms.

Yao's MINIMAX principle: proof

Proof.

$$\begin{aligned}
\min_{A_i \in \mathcal{A}} E[T(A_i, I_p)] &\leq \max_{u \in \Delta_m} \min_{A_i \in \mathcal{A}} E[T(A_i, I_u)] && (1)\\
&= \max_{u \in \Delta_m} \min_{v \in \Delta_n} E[T(A_v, I_u)] && (2)\\
&= \min_{v \in \Delta_n} \max_{u \in \Delta_m} E[T(A_v, I_u)] && (3)\\
&= \min_{v \in \Delta_n} \max_{I_j \in \mathcal{I}} E[T(A_v, I_j)] && (4)\\
&\leq \max_{I_j \in \mathcal{I}} E[T(A_q, I_j)] && (5)
\end{aligned}$$

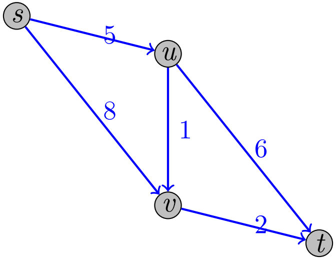
□

- Here, Δ_n denotes the set of n-dimensional probability vectors.
- Equation (3) follows by the von Neumann's MINIMAX theorem.

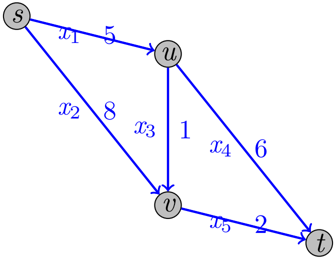
Application 4: Revisiting ShortestPath algorithm

ShortestPath problem

INPUT: n cities, and a collection of roads. A road from city i to j has a distance $d(i, j)$. Two specific cities: s and t.
OUTPUT: the shortest path from city s to t.



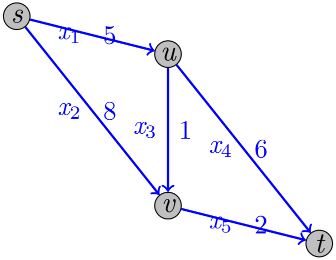
SHORTESTPATH problem: PRIMAL problem



- PRIMAL problem: set variables for **roads** (Intuition: $x_i = 0/1$ means whether edge i appears in the shortest path), and a constraint means that *"we enter a node through an edge and leaves it through another edge"*.

$$\begin{array}{lrcrcrcrcrll}
\min & 5x_1 & + & 8x_2 & + & 1x_3 & + & 6x_4 & + & 2x_5 & & \\
s.t. & x_1 & + & x_2 & & & & & & & = 1 & \text{vertex } s \\
 & & & & & & - & x_4 & - & x_5 & = -1 & \text{vertex } t \\
 & -x_1 & & & + & x_3 & + & x_4 & & & = 0 & \text{vertex } u \\
 & & - & x_2 & - & x_3 & & & + & x_5 & = 0 & \text{vertex } v \\
 & x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & = 0/1 &
\end{array}$$

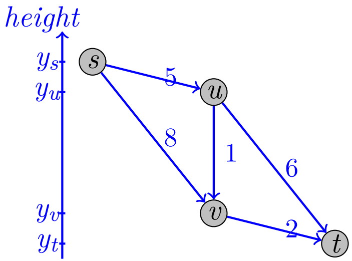
ShorestPath problem: Primal problem



- Primal problem: relax the 0/1 integer linear program into linear program by the **totally uni-modular** property.

$$
\begin{array}{rrcrcrcrcrcrll}
\min & 5x_1 & + & 8x_2 & + & 1x_3 & + & 6x_4 & + & 2x_5 & & \\
s.t. & x_1 & + & x_2 & & & & & & & =1 & \text{vertex } s \\
 & & & & & & - & x_4 & - & x_5 & =-1 & \text{vertex } t \\
 & -x_1 & & & + & x_3 & + & x_4 & & & =0 & \text{vertex } u \\
 & & - & x_2 & - & x_3 & & & + & x_5 & =0 & \text{vertex } v \\
 & x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & \geq 0 & \\
 & x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & \leq 1 &
\end{array}
$$

ShortestPath problem: Dual problem



- Dual problem: set variables for **cities**. (Intuition: y_i means the height of city i; thus, $y_s - y_t$ denotes the height difference between s and t, providing a lower bound of the shortest path length.)

$$\begin{array}{rrrrrrrrrl}
\max & y_s & - & y_t & & & & & & \\
s.t. & y_s & & & - & y_u & & & \leq 5 & x_1 : \text{edge } (s, u) \\
 & y_s & & & & & - & y_v & \leq 8 & x_2 : \text{edge } (s, v) \\
 & & & & & y_u & - & y_v & \leq 1 & x_3 : \text{edge } (u, v) \\
 & & - & y_t & + & y_u & & & \leq 6 & x_4 : \text{edge } (u, t) \\
 & & - & y_t & & & + & y_v & \leq 2 & x_5 : \text{edge } (v, t)
\end{array}$$

DUAL SIMPLEX method

Revisiting Primal Simplex algorithm

- Consider the following Primal problem **P**:

$$\begin{array}{rcccccccc} \min & x_1 & + & 14x_2 & + & 6x_3 & & \\ s.t. & x_1 & + & x_2 & + & x_3 & \leq & 4 \\ & x_1 & & & & & \leq & 2 \\ & & & & & x_3 & \leq & 3 \\ & & & 3x_2 & + & x_3 & \leq & 6 \\ & x_1 & , & x_2 & , & x_3 & \geq & 0 \end{array}$$

- Primal simplex tabular:

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
-z= 0	$\overline{c_1}=1$	$\overline{c_2}$=14	$\overline{c_3}$=6	$\overline{c_4}$=0	$\overline{c_5}$=0	$\overline{c_6}$=0	$\overline{c_7}$=0
$\mathbf{x_{B1}} = b'_1$=4	1	1	1	1	0	0	0
$\mathbf{x_{B2}} = b'_2$=2	1	0	0	0	1	0	0
$\mathbf{x_{B3}} = b'_3$=3	0	0	1	0	0	1	0
$\mathbf{x_{B4}} = b'_4$=6	0	3	1	0	0	0	1

- Primal variables: $\mathbf{x}$; Feasible: $\mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$.
- A basis $\mathbf{B}$ is called **primal feasible** if all elements in $\mathbf{B}^{-1}\mathbf{b}$ (the first column except for $-z$) are non-negative.

Revisiting PRIMAL SIMPLEX algorithm cont'd

- Now let's consider the DUAL problem **D**:

$$\begin{array}{rccccccccc} \max & 4y_1 & + & 2y_2 & + & 3y_3 & + & 6y_4 & & \\ s.t. & y_1 & + & y_2 & & & & & \leq & 1 \\ & y_1 & & & & & + & 3y_4 & \leq & 14 \\ & y_1 & & & + & y_3 & + & y_4 & \leq & 6 \\ & y_1 & , & y_2 & , & y_3 & , & y_4 & \leq & 0 \end{array}$$

- PRIMAL simplex tabular:

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
-z= 0	$\overline{c_1}$= 1	$\overline{c_2}$=14	$\overline{c_3}$=6	$\overline{c_4}$=0	$\overline{c_5}$=0	$\overline{c_6}$=0	$\overline{c_7}$=0
$\mathbf{x_{B1}} = b'_1$=4	1	1	1	1	0	0	0
$\mathbf{x_{B2}} = b'_2$=2	1	0	0	0	1	0	0
$\mathbf{x_{B3}} = b'_3$=3	0	0	1	0	0	1	0
$\mathbf{x_{B4}} = b'_4$=6	0	3	1	0	0	0	1

- Dual variables: $\mathbf{y^T} = \mathbf{c_B^T B^{-1}}$; Feasible: $\mathbf{y^T A} \leq \mathbf{c^T}$.
- A basis $\mathbf{B}$ is called **dual feasible** if all elements in $\overline{\mathbf{c^T}} = \mathbf{c} - \mathbf{c_B^T B^{-1} A} = \mathbf{c^T} - \mathbf{y^T A}$ (the first row except for $-z$) are non-negative.

Another view point of the PRIMAL SIMPLEX algorithm

- Thus another view point of the PRIMAL SIMPLEX algorithm can be described as:
  1. **Starting point:** The PRIMAL SIMPLEX algorithm starts with a primal feasible solution $(\mathbf{x_B} = \mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0})$;
  2. **Maintenance:** Throughout the process we maintain the primal feasibility and move towards the dual feasibility;
  3. **Stopping criteria:** $\overline{\mathbf{c}}^{\mathbf{T}} = \mathbf{c}^{\mathbf{T}} - \mathbf{c}_{\mathbf{B}}^{\mathbf{T}}\mathbf{B}^{-1}\mathbf{A} \geq \mathbf{0}$, i.e., $\mathbf{y}^{\mathbf{T}}\mathbf{A} \leq \mathbf{c}^{\mathbf{T}}$. In other words, the iteration process ends when the basis is both primal feasible and dual feasible.

Dual simplex works in just an opposite fashion

- Dual simplex:
  1. **Starting point:** The Dual Simplex algorithm starts with a dual feasible solution ($\overline{\mathbf{c}}^{\mathbf{T}} \geq \mathbf{0}$);
  2. **Maintenance:** Throughout the process we maintain the dual feasibility and move towards the dual feasibility;
  3. **Stopping criteria:** $\mathbf{x_B} = \mathbf{B}^{-1}\mathbf{b} \geq \mathbf{0}$. In other words, the iteration process ends when the basis is both primal feasible and dual feasible.

Primal simplex vs. Dual simplex

- Both PRIMAL SIMPLEX and DUAL SIMPLEX terminate at the same condition, i.e. the basis is both primal feasible and dual feasible.
- However, the final objective is achieved in totally opposite fashions— the PRIMAL SIMPLEX method keeps the primal feasibility while the DUAL SIMPLEX method keeps the dual feasibility during the pivoting process.
- The PRIMAL SIMPLEX algorithm *first selects an entering variable and then determines the leaving variable*.
- In contrast, the DUAL SIMPLEX method does the opposite; it *first selects a leaving variable and then determines an entering variable*.

```
DUAL SIMPLEX($B_I, z, \mathbf{A}, \mathbf{b}, \mathbf{c}$)
 1: //DUAL SIMPLEX starts with a dual feasible basis. Here, $B_I$ contains the
    indices of the basic variables.
 2: while TRUE do
 3:     if there is no index $l$ $(1 \le l \le m)$ has $b_l < 0$ then
 4:         $\mathbf{x}$ =CALCULATEX($B_I, \mathbf{A}, \mathbf{b}, \mathbf{c}$);
 5:         return $(\mathbf{x}, z)$;
 6:     end if;
 7:     choose an index $l$ having $b_l < 0$ according to a certain rule;
 8:     for each index $j$ $(1 \le i \le n)$ do
 9:         if $a_{lj} < 0$ then
10:             $\Delta_j = -\frac{c_j}{a_{lj}}$;
11:         else
12:             $\Delta_j = \infty$;
13:         end if
14:     end for
15:     choose an index $e$ that minimizes $\Delta_j$;
16:     if $\Delta_e = \infty$ then
17:         return ``no feasible solution'';
18:     end if
19:     $(B_I, \mathbf{A}, \mathbf{b}, \mathbf{c}, z)$ = PIVOT($B_I, \mathbf{A}, \mathbf{b}, \mathbf{c}, z, e, l$);
20: end while
```

An example

- Standard form:

$$\begin{array}{rrcrcrcr}
\min & 5x_1 & + & 35x_2 & + & 20x_3 & & \\
s.t. & x_1 & - & x_2 & - & x_3 & \leq & -2 \\
 & -x_1 & - & 3x_2 & & & \leq & -3 \\
 & x_1 & , & x_2 & , & x_3 & \geq & 0
\end{array}$$

- Slack form:

$$\begin{array}{rrcrcrcrcrcr}
\min & 5x_1 & + & 35x_2 & + & 20x_3 & & & & & & \\
s.t. & x_1 & - & x_2 & - & x_3 & + & x_4 & & & = & -2 \\
 & -x_1 & - & 3x_2 & & & & & + & x_5 & = & -3 \\
 & x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & \geq & 0
\end{array}$$

Step 1

	x_1	x_2	x_3	x_4	x_5
-z= 0	$\overline{c_1}$= 5	$\overline{c_2}$=35	$\overline{c_3}$=20	$\overline{c_4}$=0	$\overline{c_5}$=0
$\mathbf{x_{B1}} = b'_1$=-2	1	-1	-1	1	0
$\mathbf{x_{B2}} = b'_2$=-3	-1	-3	0	0	1

- Basis (in blue): $\mathbf{B} = \{\mathbf{a_4}, \mathbf{a_5}\}$
- Solution: $\mathbf{x} = \begin{bmatrix} \mathbf{B^{-1}b} \\ \mathbf{0} \end{bmatrix} = [0, 0, 0, -2, -3]^T$.
- Pivoting: choose $\mathbf{a_5}$ to leave basis since $b'_2 = -3 < 0$; choose $\mathbf{a_1}$ to enter basis since $\min_{j, a_{2j}<0} \frac{\overline{c}_j}{-a_{2j}} = \frac{\overline{c}_1}{-a_{21}}$.

Step 2

	x_1	x_2	x_3	x_4	x_5
-z= -15	$\overline{c_1}$= 0	$\overline{c_2}$=20	$\overline{c_3}$=20	$\overline{c_4}$=0	$\overline{c_5}$=5
$\mathbf{x_{B1}} = b'_1$=-5	0	-4	-1	1	1
$\mathbf{x_{B2}} = b'_2$=3	1	3	0	0	-1

- Basis (in blue): $\mathbf{B} = \{\mathbf{a_1}, \mathbf{a_4}\}$
- Solution: $\mathbf{x} = \begin{bmatrix} \mathbf{B}^{-1}\mathbf{b} \\ \mathbf{0} \end{bmatrix} = [3, 0, 0, -5, 0]^T$.
- Pivoting: choose $\mathbf{a_4}$ to leave basis since $b'_1 = -5 < 0$; choose $\mathbf{a_2}$ to enter basis since $\min_{j, a_{1j}<0} \frac{\overline{c}_j}{-a_{1j}} = \frac{\overline{c}_2}{-a_{12}}$.

Step 3

	x_1	x_2	x_3	x_4	x_5
-z= -40	$\overline{c_1}=0$	$\overline{c_2}=0$	$\overline{c_3}=15$	$\overline{c_4}=5$	$\overline{c_5}=10$
$\mathbf{x_{B1}} = b_1' = \frac{5}{4}$	0	1	$\frac{1}{4}$	$-\frac{1}{4}$	$-\frac{1}{4}$
$\mathbf{x_{B2}} = b_2' = -\frac{3}{4}$	1	0	$-\frac{3}{4}$	$\frac{3}{4}$	$-\frac{1}{4}$

- Basis (in blue): $\mathbf{B} = \{\mathbf{a_1}, \mathbf{a_2}\}$
- Solution: $\mathbf{x} = \begin{bmatrix} \mathbf{B}^{-1}\mathbf{b} \\ \mathbf{0} \end{bmatrix} = [\frac{5}{4}, -\frac{3}{4}, 0, 0, 0]^T$.
- Pivoting: choose $\mathbf{a_1}$ to leave basis since $b_2' = -\frac{3}{4} < 0$; choose $\mathbf{a_3}$ to enter basis since $\min_{j, a_{2j}<0} \frac{\bar{c}_j}{-a_{2j}} = \frac{\bar{c}_3}{-a_{23}}$.

Step 4

	x_1	x_2	x_3	x_4	x_5
-z= -55	$\overline{c_1}$= 20	$\overline{c_2}$=0	$\overline{c_3}$=0	$\overline{c_4}$=20	$\overline{c_5}$=5
$\mathbf{x_{B1}} = b'_1$=1	$\frac{1}{3}$	1	0	0	$-\frac{1}{3}$
$\mathbf{x_{B2}} = b'_2$=1	$-\frac{4}{3}$	0	1	-1	$\frac{1}{3}$

- Basis (in blue): $\mathbf{B} = \{\mathbf{a_2}, \mathbf{a_3}\}$
- Solution: $\mathbf{x} = \begin{bmatrix} \mathbf{B}^{-1}\mathbf{b} \\ \mathbf{0} \end{bmatrix} = [0, 1, 1, 0, 0]^T$.
- Done!

When dual simplex method is useful?

1. The dual simplex algorithm is most suited for problems for which an **initial dual feasible solution** is easily available.
2. It is particularly useful for **reoptimizing** a problem after a constraint has been added or some parameters have been changed so that the previously optimal basis is no longer feasible.
3. Trying dual simplex is particularly useful if your LP appears to be highly degenerate, i.e. there are many vertices of the feasible region for which the associated basis is degenerate. We may find that a large number of iterations (moves between adjacent vertices) occur with little or no improvement.[1]

[1] References: Operations Research Models and Methods, Paul A. Jensen and Jonathan F. Bard; OR-Notes, J. E. Beasley

Primal_Dual: another IMPROVMENT approach

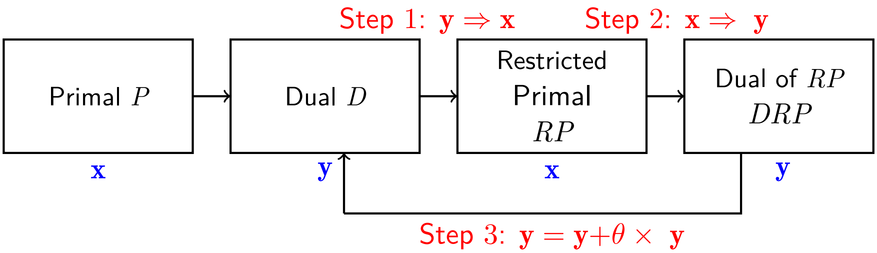
Primal_Dual method: a brief history

- In 1955, H. Kuhn proposed the Hungarian method for the MAXWEIGHTEDMATCHING problem. This method effectively explores the duality property of linear programming.
- In 1956, G. Dantzig, R. Ford, and D. Fulkerson extended this idea to solve linear programming problems.
- In 1957, R. Ford, and D. Fulkerson applied this idea to solve network-flow problem and Hitchcock problem.
- In 1957, J. Munkres applied this idea to solve the transportation problem.

Primal_Dual method: basic idea

Primal_Dual Method

- Primal_Dual method is a dual method, which exploits the lower bound information in subsequent linear programming operations.
- Advantages:
  1. Unlike dual simplex starting from a **dual basic feasible solution**, primal_dual method requires only a **dual feasible solution**.
  2. An optimal solution to DRP usually has combinatorial explanation, especially for graph-theory problems.


Step 1: $\mathbf{y} \Rightarrow \mathbf{x}$
Step 2: $\mathbf{x} \Rightarrow \mathbf{y}$
Primal P
Dual D
Restricted Primal RP
Dual of RP DRP
$\mathbf{x}$
$\mathbf{y}$
$\mathbf{x}$
$\mathbf{y}$
Step 3: $\mathbf{y} = \mathbf{y} + \theta \times \mathbf{y}$

Basic idea of primal_dual method

- Primal P:

$$\begin{array}{rcccccccccl}
\min & c_1x_1 & + & c_2x_2 & + & \dots & + & c_nx_n & & & \\
s.t. & a_{11}x_1 & + & a_{12}x_2 & + & \dots & + & a_{1n}x_n & = & b_1 & (y_1) \\
 & a_{21}x_1 & + & a_{22}x_2 & + & \dots & + & a_{2n}x_n & = & b_2 & (y_2) \\
 & & & & & \dots & & & & & \\
 & a_{m1}x_1 & + & a_{m2}x_2 & + & \dots & + & a_{mn}x_n & = & b_m & (y_m) \\
 & x_1 & , & x_2 & , & \dots & , & x_n & \geq & 0 &
\end{array}$$

- Dual D:

$$\begin{array}{rccccccccc}
\max & b_1y_1 & + & b_2y_2 & + & \dots & + & b_my_m & & \\
s.t. & a_{11}y_1 & + & a_{21}y_2 & + & \dots & + & a_{m1}y_m & \leq & c_1 \\
 & a_{12}y_1 & + & a_{22}y_2 & + & \dots & + & a_{m2}y_m & \leq & c_2 \\
 & & & & & \dots & & & & \\
 & a_{1n}y_1 & + & a_{2n}y_2 & + & \dots & + & a_{mn}y_m & \leq & c_n
\end{array}$$

- Basic idea: Suppose we are given a dual feasible solution $\mathbf{y}$. Let's verify whether $\mathbf{y}$ is an optimal solution or not:
  1. If $\mathbf{y}$ is an optimal solution to the dual problem D, then the corresponding primal variables $\mathbf{x}$ should satisfy a restricted primal problem called RP;
  2. Furthermore, even if $\mathbf{y}$ is not optimal, the solution to the dual of RP (called DRP) still provide invaluable information — it can be used to improve $\mathbf{y}$.

Step 1: $\mathbf{y} \Rightarrow \mathbf{x}$ I

- Dual problem D:

$$\begin{array}{lccccccccccl}
\max & b_1 y_1 & + & b_2 y_2 & + & \ldots & + & b_m y_m & & & \\
s.t. & a_{11} y_1 & + & a_{21} y_2 & + & \ldots & + & a_{m1} y_m & \leq & c_1 & ('=' \Rightarrow x_1 \geq 0) \\
 & & & & & \ldots & & & & & \\
 & a_{1n} y_1 & + & a_{2n} y_2 & + & \ldots & + & a_{mn} y_m & \leq & c_n & ('<' \Rightarrow x_n = 0)
\end{array}$$

- $\mathbf{y}$ provides information of the corresponding primal variables $\mathbf{x}$:
  1. Given a dual feasible solution $\mathbf{y}$. Let's check whether $\mathbf{y}$ is optimal solution or not.
  2. If $\mathbf{y}$ is optimal, we have the following restrictions on $\mathbf{x}$:
     $a_{1i}y_1 + a_{2i}y_2 + \ldots + a_{mi}y_m < c_i \Rightarrow x_i = 0$
     (Reason: complement slackness. An optimal solution $\mathbf{y}$ satisfies $(a_{1i}y_1 + a_{2i}y_2 + \ldots + a_{mi}y_m - c_i) \times x_i = 0)$
  3. Let's use J to record the index of **tight constraints** where "$=$" holds.

Step 1: $\mathbf{y} \Rightarrow \mathbf{x}$ II

$$\begin{array}{ccc|c}
 & J & & x_n = 0 \\
 & & & \Uparrow \\
c_1 & c_2 & \cdots & c_n \\
\| & \| & & \vee \\
y_1 a_{11} & y_1 a_{12} & \cdots & y_1 a_{1n} \\
+ & + & & + \\
y_2 a_{21} & y_2 a_{22} & \cdots & y_2 a_{2n} \\
+ & + & & + \\
\vdots & \vdots & \cdots & \vdots \\
+ & + & & + \\
y_m a_{m1} & y_m a_{m2} & \cdots & y_m a_{mn}
\end{array}$$

Step 1: $\mathbf{y} \Rightarrow \mathbf{x}$ III

4. Thus the corresponding primal solution $\mathbf{x}$ should satisfy the following restricted primal (RP):
5. RP:

$$\begin{array}{ccccccccccc}
a_{11}x_1 & + & a_{12}x_2 & + & ... & + & a_{1n}x_n & = & b_1 & \\
a_{21}x_1 & + & a_{22}x_2 & + & ... & + & a_{2n}x_n & = & b_2 & \\
 & & & & ... & & & & & \\
a_{m1}x_1 & + & a_{m2}x_2 & + & ... & + & a_{mn}x_n & = & b_m & \\
 & & & & & & x_i & = & 0 & i \notin J \\
 & & & & & & x_i & \geq & 0 & i \in J
\end{array}$$

6. In other words, the optimality of $\mathbf{y}$ is determined via solving RP.

But how to solve RP? I

- RP:

$$\begin{array}{ccccccccccc}
a_{11}x_1 & + & a_{12}x_2 & + & ... & + & a_{1n}x_n & = & b_1 & \\
a_{21}x_1 & + & a_{22}x_2 & + & ... & + & a_{2n}x_n & = & b_2 & \\
 & & & & ... & & & & & \\
a_{m1}x_1 & + & a_{m2}x_2 & + & ... & + & a_{mn}x_n & = & b_m & \\
 & & & & & & x_i & = & 0 & i \notin J \\
 & & & & & & x_i & \geq & 0 & i \in J
\end{array}$$

- How to solve RP? Recall that $\mathbf{Ax} = \mathbf{b}, \mathbf{x} \geq \mathbf{0}$ can be solved via solving an extended LP.

But how to solve RP? II

- RP (extended through introducing slack variables):

$$
\begin{array}{lllllllllll}
\min & \epsilon = & s_1 & +s_2 & \dots & +s_m & & & & & \\
s.t. & & s_1 & & & & +a_{11}x_1 & \dots & +a_{1n}x_n & = b_1 & \\
 & & & s_2 & & & +a_{21}x_1 & \dots & +a_{2n}x_n & = b_2 & \\
 & & & & \dots & & & \dots & & & \\
 & & & & & s_m & +a_{m1}x_1 & \dots & +a_{mn}x_n & = b_m & \\
 & & & & & & & & x_i & = 0 & i \notin \\
 & & & & & & & & x_i & \geq 0 & i \in \\
 & & & & & & & & s_i & \geq 0 & \forall i
\end{array}
$$

1. If $\epsilon_{OPT} = 0$, then we find a feasible solution to RP, implying that $\mathbf{y}$ is an optimal solution;
2. If $\epsilon_{OPT} > 0$, $\mathbf{y}$ is not an optimal solution.

Step 2: $\mathbf{x} \Rightarrow \mathbf{y}$ I

- Alternatively, we can solve the dual of RP, called DRP:

$$\begin{array}{llllllllllll}
\max & w & = & b_1 y_1 & + & b_2 y_2 & + & \ldots & + & b_m y_m & & \\
s.t. & & & a_{11} y_1 & + & a_{21} y_2 & + & \ldots & + & a_{m1} y_m & \leq & 0 \\
 & & & a_{12} y_1 & + & a_{22} y_2 & + & \ldots & + & a_{m2} y_m & \leq & 0 \\
 & & & & & & & \ldots & & & & \\
 & & & a_{1|J|} y_1 & + & a_{2|J|} y_2 & + & \ldots & + & a_{m|J|} y_m & \leq & 0 \\
 & & & y_1 & , & y_2 & , & & , & y_m & \leq & 1
\end{array}$$

1. If $w_{OPT} = 0$, $\mathbf{y}$ is an optimal solution
2. If $w_{OPT} > 0$, $\mathbf{y}$ is not an optimal solution. However, the optimal solution still provides useful information — the optimal solution to DRP can be used to improve $\mathbf{y}$.

The difference between DRP and D

- Dual problem D:

$$\begin{array}{rccccccccc}
\max & b_1 y_1 & + & b_2 y_2 & + & \ldots & + & b_m y_m & & \\
s.t. & a_{11} y_1 & + & a_{21} y_2 & + & \ldots & + & a_{m1} y_m & \leq & c_1 \\
 & a_{12} y_1 & + & a_{22} y_2 & + & \ldots & + & a_{m2} y_m & \leq & c_2 \\
 & & & & & \ldots & & & & \\
 & a_{1n} y_1 & + & a_{2n} y_2 & + & \ldots & + & a_{mn} y_m & \leq & c_n
\end{array}$$

- DRP:

$$\begin{array}{rcccccccccccc}
\max & w & = & b_1 y_1 & + & b_2 y_2 & + & \ldots & + & b_m y_m & & \\
s.t. & & & a_{11} y_1 & + & a_{21} y_2 & + & \ldots & + & a_{m1} y_m & \leq & 0 \\
 & & & a_{12} y_1 & + & a_{22} y_2 & + & \ldots & + & a_{m2} y_m & \leq & 0 \\
 & & & & & & & \ldots & & & & \\
 & & & a_{1|J|} y_1 & + & a_{2|J|} y_2 & + & \ldots & + & a_{m|J|} y_m & \leq & 0 \\
 & & & y_1 & , & y_2 & , & & , & y_m & \leq & 1
\end{array}$$

- How to write DRP from D?
  - Replacing c_i with 0;
  - Only $|J|$ restrictions in DRP;
  - An additional restriction: $y_1, y_2, \ldots, y_m \leq 1$;

Step 3: $\mathbf{y} \Rightarrow \mathbf{y}$ I

Why $\mathbf{y}$ can be used to improve $\mathbf{y}$? Consider an improved dual solution $\mathbf{y}' = \mathbf{y} + \theta\ \mathbf{y}, \theta > 0$. We have:

- **Objective function:** Since $\mathbf{y}^\mathbf{T}\mathbf{b} = w_{OPT} > 0$, $\mathbf{y}'^\mathbf{T}\mathbf{b} = \mathbf{y}^\mathbf{T}\mathbf{b} + \theta w_{OPT} > \mathbf{y}^\mathbf{T}\mathbf{b}$. In other words, $(\mathbf{y} + \theta\ \mathbf{y})$ is better than $\mathbf{y}$.
- **Constraints:** The dual feasibility requires that:
  - For any $j \in J$, $a_{1j}\Delta y_1 + a_{2j}\Delta y_2 + ... + a_{mj}\Delta y_m \leq 0$. Thus we have $\mathbf{y}'^\mathbf{T}\mathbf{a_j} = \mathbf{y}^\mathbf{T}\mathbf{a_j} + \theta\ \mathbf{y}^\mathbf{T}\mathbf{a_j} \leq \mathbf{c_j}$ for any $\theta > 0$.
  - For any $j \notin J$, there are two cases:

Step 3: $\mathbf{y} \Rightarrow \mathbf{y}$ II

1. $\forall j \notin J, a_{1j}\Delta y_1 + a_{2j}\Delta y_2 + \ldots + a_{mj}\Delta y_m \leq 0$:
   Thus $\mathbf{y}'$ is feasible for any $\theta > 0$ since for $\forall 1 \leq j \leq n$,

$$\begin{aligned} & a_{1j}y'_1 + a_{2j}y'_2 + \ldots + a_{mj}y'_m && (6)\\ = \; & a_{1j}y_1 + a_{2j}y_2 + \ldots + a_{mj}y_m && (7)\\ + \; & \theta(a_{1j}\Delta y_1 + a_{2j}\Delta y_2 + \ldots + a_{mj}\Delta y_m) && (8)\\ \leq \; & c_j && (9)\end{aligned}$$

   Hence dual problem D is unbounded and the primal problem P is infeasible.
2. $\exists j \notin J, a_{1j}\Delta y_1 + a_{2j}\Delta y_2 + \ldots + a_{mj}\Delta y_m > 0$:
   We can safely set $\theta \leq \frac{c_j - (a_{1j}y_1 + a_{2j}y_2 + \ldots + a_{mj}y_m)}{a_{1j}\Delta y_1 + a_{2j}\Delta y_2 + \ldots + a_{mj}\Delta y_m} = \frac{\mathbf{c_j} - \mathbf{y}^{\mathbf{T}}\mathbf{a_j}}{\mathbf{y}^{\mathbf{T}}\mathbf{a_j}}$ to guarantee that $\mathbf{y}'^{\mathbf{T}}\mathbf{a_j} = \mathbf{y}^{\mathbf{T}}\mathbf{a_j} + \theta\, \mathbf{y}^{\mathbf{T}}\mathbf{a_j} \leq \mathbf{c_j}$.

Primal_Dual algorithm

```
1: Infeasible = "No"
   Optimal = "No"
   $\mathbf{y} = \mathbf{y_0}$;        //$\mathbf{y_0}$ is a feasible solution to the dual problem $D$
2: while TRUE do
3:     Finding tight constraints index $J$, and set corresponding $x_j = 0$ for
       $j \notin J$.
4:     Thus we have a smaller RP.
5:     Solve DRP. Denote the solution as $\Delta\mathbf{y}$.
6:     if DRP objective function $w_{OPT} = 0$ then
7:         Optimal="Yes"
8:         return  $y$;
9:     end if
10:    if  $\mathbf{y^T a_j} \leq \mathbf{0}$ (for all $j \notin J$) then
11:        Infeasible = "Yes";
12:        return  ;
13:    end if
14:    Set $\theta = \min \frac{\mathbf{c_j - y^T a_j}}{\mathbf{y^T a_j}}$ for  $\mathbf{y^T a_j} > 0$, $j \notin J$.
15:    Update $\mathbf{y}$ as $\mathbf{y} = \mathbf{y} + \theta\ \mathbf{y}$;
16: end while
```

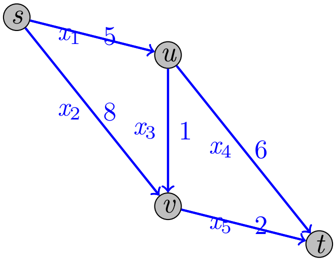
Advantages of Primal_Dual algorithm

- Some facts:
  - Primal_dual algorithm ends if using anti-cycling rule. (Reason: the objective value $\mathbf{y}^{\mathbf{T}}\mathbf{b}$ increases if there is no degeneracy.)
  - Both RP and DRP do not explicitly rely on $\mathbf{c}$. In fact, the information of $\mathbf{c}$ is represented in J.
  - This leads to another advantage of primal_dual technique, i.e., RP is usually a purely combinatorial problem. Take ShortestPath as an example. RP corresponds to a "connection" problem.
  - More and more constraints become tight in the primal_dual process.

  (See Lecture 10 for a primal_dual algorithm for MaximumFlow problem.)

SHORTESTPATH: Dijkstra's algorithm is essentially Primal_Dual algorithm

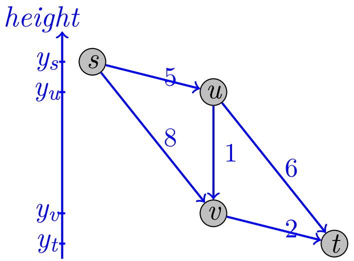
ShortestPath problem



- Primal problem: relax the 0/1 integer linear program into linear program by the **totally uni-modular** property.

$$
\begin{array}{lccccccccccll}
\min & 5x_1 & + & 8x_2 & + & 1x_3 & + & 6x_4 & + & 2x_5 & & \\
s.t. & x_1 & + & x_2 & & & & & & & = 1 & \text{vertex } s \\
 & & & & & & - & x_4 & - & x_5 & = -1 & \text{vertex } t \\
 & -x_1 & & & + & x_3 & + & x_4 & & & = 0 & \text{vertex } u \\
 & & - & x_2 & - & x_3 & & & + & x_5 & = 0 & \text{vertex } v \\
 & x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & \geq 0 & \\
 & x_1 & , & x_2 & , & x_3 & , & x_4 & , & x_5 & \leq 1 &
\end{array}
$$

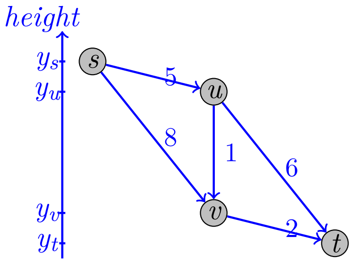
Dual of SHORTESTPATH problem



- DUAL PROBLEM: set variables for **cities**. (Intuition: y_i means the height of city i; thus, $y_s - y_t$ denotes the height difference between s and t, providing a lower bound of the shortest path length.)

$$\begin{array}{rrrrrrrrl}
\max & y_s & - & y_t & & & & & \\
s.t. & y_s & & & - & y_u & & & \leq 5 \quad x_1 : \text{edge } (s, u) \\
 & y_s & & & & & - & y_v & \leq 8 \quad x_2 : \text{edge } (s, v) \\
 & & & & & y_u & - & y_v & \leq 1 \quad x_3 : \text{edge } (u, v) \\
 & & - & y_t & + & y_u & & & \leq 6 \quad x_4 : \text{edge } (u, t) \\
 & & - & y_t & & & + & y_v & \leq 2 \quad x_5 : \text{edge } (v, t)
\end{array}$$

A simplified version



- DUAL PROBLEM: simplify by setting $y_t = 0$ (and remove the 2nd constraint in the primal problem P, accordingly)

$$
\begin{array}{lccccccl}
\max & y_s & & & & & & \\
s.t. & y_s & - & y_u & & & \leq 5 & x_1 : \text{edge } (s, u) \\
 & y_s & & & - & y_v & \leq 8 & x_2 : \text{edge } (s, v) \\
 & & & y_u & - & y_v & \leq 1 & x_3 : \text{edge } (u, v) \\
 & & & y_u & & & \leq 6 & x_4 : \text{edge } (u, t) \\
 & & & & & y_v & \leq 2 & x_5 : \text{edge } (v, t)
\end{array}
$$

Iteration 1 I

- Dual feasible solution: $\mathbf{y^T} = (\mathbf{0}, \mathbf{0}, \mathbf{0})$. Let's check the constraints in D:

$$\begin{array}{cccccccl}
y_s & - & y_u & & & < & 5 & \Rightarrow x_1 = 0 \\
y_s & & & - & y_v & < & 8 & \Rightarrow x_2 = 0 \\
 & & y_u & - & y_v & < & 1 & \Rightarrow x_3 = 0 \\
 & & y_u & & & < & 6 & \Rightarrow x_4 = 0 \\
 & & & & y_v & < & 2 & \Rightarrow x_5 = 0
\end{array}$$

- Identifying tight constraints in D: $J = \Phi$, implying that $x_1, x_2, x_3, x_4, x_5 = 0$.

- RP:

$$\begin{array}{lcccccccccl}
\min & s_1 & +s_2 & +s_3 & & & & & & & \\
s.t. & s_1 & & & +x_1 & +x_2 & & & & = 1 & \text{node } s \\
 & & s_2 & & -x_1 & & +x_3 & +x_4 & & = 0 & \text{node } u \\
 & & & s_3 & & -x_2 & -x_3 & & +x_5 & = 0 & \text{node } v \\
 & s_1, & s_2, & s_3, & & & & & & \geq 0 & \\
 & & & & x_1, & x_2, & x_3, & x_4, & x_5 & = 0 &
\end{array}$$

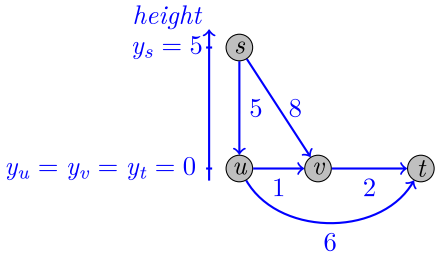
Iteration 1 II

- *DRP*:

$$\begin{array}{rllll} \max & y_s & & & \\ s.t. & y_s & & & \leq 1 \\ & & y_u & & \leq 1 \\ & & & y_v & \leq 1 \end{array}$$

- Solve *DRP* using combinatorial technique: optimal solution $\Delta\mathbf{y}^{\mathbf{T}} = (1, 0, 0)$. *Note: the optimal solution is not unique*
- Step length θ: $\theta = \min\{\frac{\mathbf{c_1} - \mathbf{y}^{\mathbf{T}}\mathbf{a_1}}{\mathbf{y}^{\mathbf{T}}\mathbf{a_1}}, \frac{\mathbf{c_2} - \mathbf{y}^{\mathbf{T}}\mathbf{a_2}}{\mathbf{y}^{\mathbf{T}}\mathbf{a_2}}\} = \min\{5, 8\} = 5$
- Update $\mathbf{y}$: $\mathbf{y}^{\mathbf{T}} = \mathbf{y}^{\mathbf{T}} + \theta\Delta\mathbf{y}^{\mathbf{T}} = (5, 0, 0)$.

Iteration 1 III



- From the point of view of Dijkstra's algorithm:
  - Optimal solution to DRP is $\Delta \mathbf{y}^{\mathbf{T}} = (1, 0, 0)$: the explored vertex set $S = \{s\}$ in Dijkstra's algorithm. In fact, DRP is solved via identifying the nodes reachable from s.
  - Step length $\theta = \min\{\frac{\mathbf{c_1} - \mathbf{y}^{\mathbf{T}}\mathbf{a_1}}{\mathbf{y}^{\mathbf{T}}\mathbf{a_1}}, \frac{\mathbf{c_2} - \mathbf{y}^{\mathbf{T}}\mathbf{a_2}}{\mathbf{y}^{\mathbf{T}}\mathbf{a_2}}\} = \min\{5, 8\} = 5$: finding the closest vertex to the nodes in S via comparing all edges going out from S.

Iteration 2 I

- Dual feasible solution: $\mathbf{y^T} = (\mathbf{5}, \mathbf{0}, \mathbf{0})$. Let's check the constraints in D:

$$\begin{array}{ccccccc} y_s & - & y_u & & & = & 5 & \\ y_s & & & - & y_v & < & 8 & \Rightarrow x_2 = 0 \\ & & y_u & - & y_v & < & 1 & \Rightarrow x_3 = 0 \\ & & y_u & & & < & 6 & \Rightarrow x_4 = 0 \\ & & & & y_v & < & 2 & \Rightarrow x_5 = 0 \end{array}$$

- Identifying tight constraints in D: $J = \{1\}$, implying that $x_2, x_3, x_4, x_5 = 0$.
- RP:

$$\begin{array}{lcccccccccl} \min & s_1 & +s_2 & +s_3 & & & & & & & \\ s.t. & s_1 & & & +x_1 & +x_2 & & & & = 1 & \text{node } s \\ & & s_2 & & -x_1 & & +x_3 & +x_4 & & = 0 & \text{node } u \\ & & & s_3 & & -x_2 & -x_3 & & +x_5 & = 0 & \text{node } v \\ & s_1, & s_2, & s_3, & & & & & & \geq 0 & \\ & & & & & x_2, & x_3, & x_4, & x_5 & = 0 & \end{array}$$

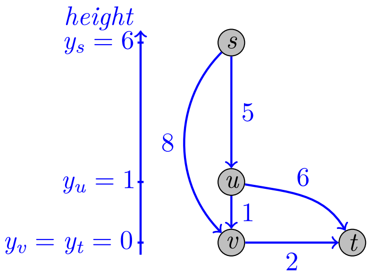
Iteration 2 II

- DRP:

$$\begin{array}{rlllll} \max & y_s & & & & \\ s.t. & y_s & - & y_u & & \leq 0 \\ & y_s & , & y_u & , \quad y_v & \leq 1 \end{array}$$

- Solve DRP using combinatorial technique: optimal solution $\Delta \mathbf{y}^{\mathbf{T}} = (1, 1, 0)$. *Note: the optimal solution is not unique*
- Step length θ: $\theta = \min\{\frac{\mathbf{c_2} - \mathbf{y}^{\mathbf{T}}\mathbf{a_2}}{\mathbf{y}^{\mathbf{T}}\mathbf{a_2}}, \frac{\mathbf{c_3} - \mathbf{y}^{\mathbf{T}}\mathbf{a_3}}{\mathbf{y}^{\mathbf{T}}\mathbf{a_3}}, \frac{\mathbf{c_4} - \mathbf{y}^{\mathbf{T}}\mathbf{a_4}}{\mathbf{y}^{\mathbf{T}}\mathbf{a_4}}\} = \min\{3, 1, 6\} = 1$
- Update $\mathbf{y}$: $\mathbf{y}^{\mathbf{T}} = \mathbf{y}^{\mathbf{T}} + \theta \Delta \mathbf{y}^{\mathbf{T}} = (6, 1, 0)$.

Iteration 2 III



- From the point of view of Dijkstra's algorithm:
  - Optimal solution to DRP is $\Delta\mathbf{y}^{\mathbf{T}} = (1, 1, 0)$: the explored vertex set $S = \{s, u\}$ in Dijkstra's algorithm. In fact, DRP is solved via identifying the nodes reachable from s.
  - Step length $\theta = \min\{\frac{\mathbf{c_2} - \mathbf{y}^{\mathbf{T}}\mathbf{a_2}}{\mathbf{y}^{\mathbf{T}}\mathbf{a_2}}, \frac{\mathbf{c_3} - \mathbf{y}^{\mathbf{T}}\mathbf{a_3}}{\mathbf{y}^{\mathbf{T}}\mathbf{a_3}}, \frac{\mathbf{c_4} - \mathbf{y}^{\mathbf{T}}\mathbf{a_4}}{\mathbf{y}^{\mathbf{T}}\mathbf{a_4}}\} = \min\{3, 1, 6\} = 1$: finding the closest vertex to the nodes in S via comparing all edges going out from S.

Iteration 3 I

- Dual feasible solution: $\mathbf{y}^{\mathbf{T}} = (\mathbf{6}, \mathbf{1}, \mathbf{0})$. Let's check the constraints in D:

$$\begin{array}{ccccccll}
y_s & - & y_u & & & = & 5 & \\
y_s & & & - & y_v & < & 8 & \Rightarrow x_2 = 0 \\
& & y_u & - & y_v & = & 1 & \\
& & y_u & & & < & 6 & \Rightarrow x_4 = 0 \\
& & & & y_v & < & 2 & \Rightarrow x_5 = 0
\end{array}$$

- Identifying tight constraints in D: $J = \{1, 3\}$, implying that $x_2, x_4, x_5 = 0$.

- RP:

$$\begin{array}{llllllllll}
\min & s_1 & +s_2 & +s_3 & & & & & & \\
s.t. & s_1 & & & +x_1 & +x_2 & & & & = 1 \quad \text{node } s \\
& & s_2 & & -x_1 & & +x_3 & +x_4 & & = 0 \quad \text{node } u \\
& & & s_3 & & -x_2 & -x_3 & & +x_5 & = 0 \quad \text{node } v \\
& s_1, & s_2, & s_3, & & & & & & \geq 0 \\
& & & & & x_2, & & x_4, & x_5 & = 0
\end{array}$$

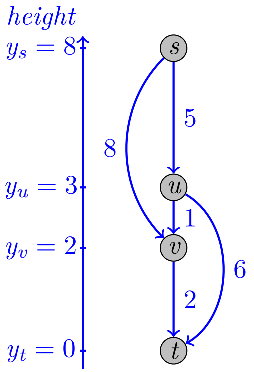
Iteration 3 II

- DRP:

$$\begin{array}{rccccccl} \max & y_s & & & & & & \\ s.t. & y_s & - & y_u & & & & \leq 0 \\ & & & y_u & - & y_v & & \leq 0 \\ & y_s & , & y_u & , & y_v & & \leq 1 \end{array}$$

- Solve DRP using combinatorial technique: optimal solution $\Delta \mathbf{y}^\mathbf{T} = (1, 1, 1)$. *Note: the optimal solution is not unique*
- Step length θ: $\theta = \min\{\frac{\mathbf{c_4} - \mathbf{y}^\mathbf{T}\mathbf{a_4}}{\mathbf{y}^\mathbf{T}\mathbf{a_4}}, \frac{\mathbf{c_5} - \mathbf{y}^\mathbf{T}\mathbf{a_5}}{\mathbf{y}^\mathbf{T}\mathbf{a_5}}\} = \min\{5, 2\} = 2$
- Update $\mathbf{y}$: $\mathbf{y}^\mathbf{T} = \mathbf{y}^\mathbf{T} + \theta \Delta \mathbf{y}^\mathbf{T} = (8, 3, 2)$.

Iteration 3 III



- From the point of view of Dijkstra's algorithm:
  - Optimal solution to DRP is $\Delta\mathbf{y}^\mathbf{T} = (1, 1, 1)$: the explored vertex set $S = \{s, u, v\}$ in Dijkstra's algorithm. In fact, DRP is solved via identifying the nodes reachable from s.

Iteration 3 IV

- Step length $\theta = \min\{\frac{\mathbf{c_4 - y^T a_4}}{\mathbf{y^T a_4}}, \frac{\mathbf{c_5 - y^T a_5}}{\mathbf{y^T a_5}}\} = \min\{5, 2\} = 2$: finding the closest vertex to the nodes in S via comparing all edges going out from S.

Iteration 4 I

- Dual feasible solution: $\mathbf{y^T} = (\mathbf{8}, \mathbf{3}, \mathbf{2})$. Let's check the constraints in D:

$$\begin{array}{ccccccccl}
y_s & & - & y_u & & & = & 5 & \\
y_s & & & & - & y_v & < & 8 & \Rightarrow x_2 = 0 \\
 & & & y_u & - & y_v & = & 1 & \\
 & & & y_u & & & < & 6 & \Rightarrow x_4 = 0 \\
 & & & & & y_v & = & 2 &
\end{array}$$

- Identifying tight constraints in D: $J = \{1, 3, 5\}$, implying that $x_2, x_4 = 0$.

- RP:

$$\begin{array}{lcccccccccl}
\min & s_1 & +s_2 & +s_3 & & & & & & & \\
s.t. & s_1 & & & +x_1 & +x_2 & & & & = 1 & \text{node } s \\
 & & s_2 & & -x_1 & & +x_3 & +x_4 & & = 0 & \text{node } u \\
 & & & s_3 & & -x_2 & -x_3 & & +x_5 & = 0 & \text{node } v \\
 & s_1, & s_2, & s_3, & & & & & & \geq 0 & \\
 & & & & & x_2, & & x_4 & & = 0 &
\end{array}$$

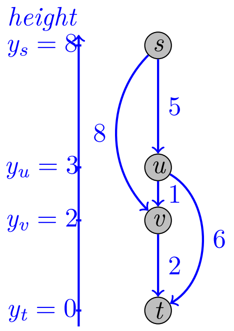
Iteration 4 II

- *DRP*:

$$\begin{array}{rccccccl} \max & y_s & & & & & & \\ s.t. & y_s & - & y_u & & & \leq 0 \\ & & & y_u & - & y_v & \leq 0 \\ & & & & & y_v & \leq 0 \\ & y_s & , & y_u & , & y_v & \leq 1 \end{array}$$

- Solve *DRP* using combinatorial technique: optimal solution $\Delta\mathbf{y}^{\mathbf{T}} = (0, 0, 0)$. Done!

Iteration 4 III



- From the point of view of Dijkstra's algorithm:
    - Optimal solution to DRP is $\Delta\mathbf{y}^{\mathbf{T}} = (0, 0, 0)$: there is a path from s to t, forcing $y_s = 0$ (note y_t is fixed to be 0). This corresponds to the explored node set $S = \{s, u, v, t\}$ in Dijkstra's algorithm.
- Another intuitive explanation: the **tightest** rope when picking up s.