Notes: Miscellanea

Michael Van Wickle

Spring 2016

Contents

1 AC Power

- Real/Active Power, P (Watts, W)
  - power delivered to a purely resistive load
- Reactive Power, Q (reactive volt-amperes, var)
  - exists in an AC circuit when voltage and current are not in phase
  - indicative of a capacitance and/or inductance in the circuit
  - $Q = V_{rms} I_{rms} sin(\phi)$
- Complex Power, S (volt-ampere, VA)
  - $S = P + jQ$
- Apparent Power, $|S|$ (volt-ampere, VA)
  - the magnitude of the complex power
- Phase of voltage relative to current, ϕ
  - if ϕ is in Quadrant I: current lagging voltage
  - if ϕ is in Quadrant IV: current leading voltage
- The unit for all kinds of power is the Watt, W
  - this is generally reserved to refer to active power
- On the imaginary plane:
  - Active power is on the real axis
  - Reactive power is on the imaginary axis, as it does no work
  - Complex power is the vector sum of the active and reactive power
  - Apparent power is the magnitude of complex power
  - The angle formed by the active and complex power vectors is ϕ
- Capacitive Loads
  - "source" reactive power
  - cause current to lead voltage
- Inductive Loads
  - "sink" reactive power
  - cause current to lag behind voltage
- Power Factor, PF
  - the ratio of real power to apparent power
  - $PF = cos\phi = \dfrac{P}{|S|}$

1.1 Three Phase Power

- Three-Phase Power
  - three conductors carry AC power of the same frequency and voltage amplitude
    - each is phase shifted one third of a period (120 degrees)

2 Algorithms

- In 2000, IEEE published a list of the 10 most important, influential algorithms of the 20^{th} century
- They are, in no particular order:
  - Metropolis Algorithm for Monte Carlo
  - Simplex Method for Linear Programming
  - Krylov Subspace Iteration Methods
  - The Decompositional Approach to Matrix Computations
  - The Fortran Optimizing Compiler
  - QR Algorithm for Computing Eigenvalues
  - Quicksort Algorithm for Sorting
  - Fast Fourier Transform
  - Integer Relation Detection
  - Fast Multipole Method

2.1 MD5

A cryptographic hashing function, defined in `RFC 1321`, it produces a 128-bit hash value, or digest. This it typically expressed as a 32 digit hexadecimal number. Commonly used to verify data integrity.

2.1.1 Definitions

- Word: 32-bits
- Byte: 8-bits
- Big Endian: Most significant byte at lowest address, least significant byte at highest address
- Little Endian: Most significant byte at highest address, least significant byte at lowest address

2.1.2 Algorithm

- Input message is an arbitrary length sequence of bits
- This is broken up into 512-bit blocks, i.e., 16 32-bit words
- The input is padded to make it divisible by 32
  1. A single 1 is appended
  2. Zeroes are added until the length is 64 bits less than a multiple of 512
  3. The remaining 64 bits are the binary representation of the original length of the input
- The main algorithm operates on a 128-bit state, divided into 4 32-bit words: A, B, C, D
- The words are initialized to constants
  - Values are: Little Endian, Hex
  - $A : 67\ \ 45\ \ 23\ \ 01$
  - $B : EF\ \ CD\ \ AB\ \ 89$
  - $C : 98\ \ BA\ \ DC\ \ FE$
  - $D : 10\ \ 32\ \ 54\ \ 76$
- They are then operated on by each 512 bit block by four different functions
  - $F(B, C, D) = (B \cdot C) + (B' \cdot D)$
  - $G(B, C, D) = (B \cdot D) + (C \cdot D')$
  - $H(B, C, D) = B \oplus C \oplus D$

- $I(B, C, D) = C \oplus (B + D')$

- For the actual algorithm, a table, `T` is constructed, consisting of 64 elements
  - for `i` from `1` to `64`:

    `T[i]=int(`2^{32}`·abs(sin(`$i+1$`))`
    - where `i` is in radians and `int()` takes the integer portion of a number
- Each *chunk*(512-bits) of the input is broken into 16 32-bit words: $M[j], 0 \leq j < 16$

2.2 Advanced Encryption Standard (AES), Rijndael

A Federal Information Processing Standard (FIPS) approved cryptographic algorithm, used to transmit secure, government data. A symmetric block cipher that can encrypt (encipher) and decrypt (decipher) information. Processes data blocks of 128-bits.

2.3 RSA

A public-key encryption system, used for secure data transmission.

- Four steps to the algorithm:
  1. Key Generation
  2. Key Distribution
  3. Encryption
  4. Decryption

2.3.1 Key Generation

1. Choose two distinct primes, p and q
   - p and q should be chosen at random and should be similar in magnitude, but differ in length by a few digits
2. Compute: $n = p \cdot q$
   - n is used as the *modulus* for both the public and private keys
   - its length, in bits, is the key length
3. Compute $\phi(n) = \phi(p)\phi(q) = (p-1)(q-1) = n - (p+q-1)$
   - $\phi(n)$ is Euler's Totient Function
     - counts the number of positive integers coprime with n
   - this value is kept private
4. Choose an integer e, such that: $1 < e < \phi(n)$ and $gcd(e, \phi(n)) = 1$
   - i.e., e and $\phi(n)$ are coprime
   - e is released as the *public key exponent*
5. Determine d such that: $d \equiv e^{-1}\ (mod\ \phi(n))$
   - d is the modular multiplicative inverse of e ($modulo\ \phi(n)$
   - i.e., solve for d, given $d \cdot e \equiv 1\ (mod\ \phi(n))$
   - d is kept as the private key exponent

- The *public key* consists of the modulus n, and the public (or encryption) exponent e
- The *private key* consists of the modulus n, and the private (or decryption) exponent d
- d, p, q, and $\phi(n)$ must all be kept secret
  - p, q, and $\phi(n)$ can be used to calculate d

2.3.2 Key Distribution

The public key, (n, e) is then transmitted to parties wishing to send encrypted messages. The private key, d, is kept secret and never revealed.

2.3.3 Encryption

- A message, M, is to be sent.
- M is turned into an integer, m, such that: $0 \leq m < n$ and $gcd(m, n) = 1$
  - m is computed using a padding scheme
    - the padding scheme is a mutually agreed upon, reversible protocol
    - e.g., Optimal Asymmetric Encryption Padding (RFC 2437)
- The cyphertext, c is computed, using the public key, e
  - $c \equiv m^e \ (mod\ n)$
- c is then transmitted

2.3.4 Decryption

- m can be recovered from c using the private key d
  - $c^d \equiv (m^e)^d \equiv m\ mod\ n$
- with m, M can be recovered by reversing the padding scheme

2.3.5 Diffie-Hellman Key Exchange

- A secure way of sending encryption keys across parties

2.4 Huffman Coding

A technique for lossless data compression. Generates a series of prefix codes which are sent in lieu of the actual characters.

2.4.1 Compression

- In this method, a binary tree of nodes is created
- There are two different types of nodes, *leaf nodes* and *internal nodes*
- Leaf Nodes contain:
  - the symbol itself
  - the weight (frequency of appearance) of the symbol
  - optionally, a link to a parent node
- Internal Nodes contain:
  - symbol weight
    - generally the sum of the weights of its children
  - links to two child nodes
    - commonly, bit 0 represents the left child and bit 1 represents the right child
  - optionally, a link to a parent node
- The process starts with the leaf nodes representing all of the symbols and their probabilities
- A new, internal, node is added, whose children are the two nodes with the smallest probability
- This new node's probability is equal to the sum of its children's probability
- The two nodes whose parent was just added are no longer considered, with their parent now being considered

• This process is repeated until an entire tree is formed

There are two general methods to constructing the tree, using a priority queue and using two queues. The priority queue method runs in logarithmic time, $O(nlog(n))$, while the method using two queues can run in linear time, $O(n)$. However, the two queue method also requires a sort prior to commencing, which is generally $O(nlog(n))$. The two methods end up being similarly complex in a time sense. In addition, the n is normally fairly small, rendering time complexity of little importance.

2.4.2 Using a Priority Queue

• In the priority queue, the lowest probability is given the highest priority
• runs in logarithmic time, $O(nlog(n))$

1. Create a leaf node for each symbol and add it to the queue
2. While there is more than one node in the queue
    1. Remove the two nodes of highest priority (lowest probability) from the queue
    2. Create a new internal node with these two nodes as children and probability equal two their sum
    3. Add the new node to the queue
3. The remaining node is the root node, tree is complete

2.4.3 Using Two Queues

• If the symbols are sorted via probability, this runs in linear time

1. Start with as many leaves as there are symbols
2. Add all nodes into the first queue, with the least likely (lowest probability) node at the front of the queue
3. While there is more than one node in the queues
    1. Remove the two nodes with the lowest weights from the queues
    2. Create an internal node with the two nodes as children and weight being equal to their sum
    3. Add the new node to the back of the second queue
4. The remaining node is the root node, tree complete

• Break ties by choosing the node in the first queue

2.4.4 Decompression

Transmit a series of prefix codes. Assuming the receiver has knowledge of the structure of the tree, which can be sent beforehand or with the codes, it traverses the tree and finds the character at each leaf.

2.5 LZ77

A technique for lossless data compression. A dictionary coder, it maintains a sliding window during compression. Generates output similar to run-length encoding.

2.6 DEFLATE

A technique for lossless data compression, defined in `RFC 1951`. It uses a combination of Huffman coding and LZ77.

2.7 Discrete Cosine Transform (DCT)

2.8 Metropolis Algorithm for Monte Carlo

Also known as the Metropolis-Hastings algorithm, it is a Markov chain Monte Carlo (MCMC) method for generating a sequence of random samples from a probability distribution in which direct sampling is difficult. Used to approximate a distribuion.

2.9 Simplex Method for Linear Programming

A linear programming problem is generally defined as the problem of maximizing or minimizing a linear function, subject to linear constraints.

2.10 Krylov Subspace Iteration Methods

2.11 The Decompositional Approach to Matrix Computations

2.12 The Fortran Optimizing Compiler

The first optimizing compiler to be programmed. It has influenced most, if not all, modern compilers.

2.13 QR Algorithm for Computing Eigenvalues

An algorithm used to find the eigenvalues and eigenvectors of a matrix.

2.13.1 QR Decomposition

A matrix decomposition of a matrix A into $A = QR$, where Q is an orthogonal matrix and R is an upper triangular matrix.

- For a square matrix
  - a real matrix, A may be decomposed as $A = QR$
  - if A is invertible, the it has a unique factorization if the diagonal elements of R need to be positive
  - if A is complex, then there is a decomposition $A = QR$, where Q is a unitary matrix ($Q^*Q = I$)
  - if A has n linearly independent columns:
    - the first n columns of Q form an orthonormal basis for the column space of A
      - generally: the first k columns of Q form an orthonormal basis for the span of the first k columns of A for any $1 \leq k \leq n$
      - the fact that any column k of A only depends on the first k columns of Q is responsible for the triangular form of R
- For a rectangular matrix
  - for a complex matrix, A, of dimensions, $m \times n$, and $m \geq n$, A can be factored as the product of an $m \times m$ unitary matrix Q and an $m \times n$ upper rectangular matrix R
  - as the bottom $(m - n)$ rows of an $m \times n$ upper rectangular matrix consists of zeroes, R or both R and Q be partitioned as:

$$A = QR = \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = \begin{bmatrix} Q_1, Q_2 \end{bmatrix} \begin{bmatrix} R_1 \\ 0 \end{bmatrix} = Q_1 R_1$$

where R_1 is an nxn upper triangular matrix, 0 is an $(m - n) \times n$ zero matrix, Q_1 is $m \times n$, Q_2 is $m \times (m - n)$, and Q_1 and Q_2 both have orthogonal columns

2.14 Quicksort Algorithm for Sorting

Generally the most efficient method of sorting, with an average complexity of $O(nlog(n))$. A comparison sort, it is generally not stable, and can be performed in place.

2.15 Fast Fourier Transform (FFT)

A method for computing the Discrete Fourier Transform (DFT) of a sequence, or its inverse.

2.16 Integer Relation Detection

An integer relation between a set of real numbers, $x_1, x_2, \ldots, x_n$ is the set of integers, $a_1, a_2, \ldots, a_n$, not all zero, such that: $a_1x_1 + a_2x_2 + \ldots + a_nx_n = 0$. Integer relation detection algorithms find these relations, or find that, given a upper bound of magnitude, no integer relation exists.

2.17 Fast Multipole Method

A method developed to speed up the calculation of long-ranged forces in the n-body problem.

3 Matrix Decompositions

3.1 Cholesky

3.2 LU

3.2.1 LU with Full Pivoting

3.3 Block LU

3.4 QR

3.5 Spectral

3.6 Schur

3.7 Singular Value

4 Elementary Number Theory

- the integer n divides the integer a if and only if there exists an integer d such that $a = nd$
  - thus, a is divisible by n, n is a divisor or factor of a, and a is a multiple of n
  - the notation $n|a$ means that n divides a, and that $n|a-b$ means that n divides $(a-b)$
- if $mn|a$, then $m|a$ and $n|a$ for any integers m and n
- a prime number is an integer greater than one, which only has positive integer divisors of one and itself
  - a non-prime number greate than one is called a composite number
- The Fundamental Theorem of Arithmetic: every integer $n > 1$ can be represented in exactly one way as a product of primes except for the order of the factors
  - i.e., n can be uniquely written in the form: $n = p_1^{k_1} \cdot p_2^{k_2} \cdot \ldots \cdot p_r^{k_r}$ for primes $p_1 < p_2 < \cdot < p_r$ and positive integers $k_1, \ldots, k_r$
- two numbers a and b which have no common factors other than one are said to be coprime or relatively prime
- the greatest common divisor of two integers a and b is the largest integer that divides both numbers
- a and b are coprime if and only if $gcd(a, b) = 1$
- *mod* as a binary operation: $a = b \ mod \ n$, defines an operation by which a is equal to the remainder of dividing b by n, $0 \leq a < n$
- *mod* as a congruence relation: $a \equiv b \ (mod \ n)$ means that a and b have the same remainder when dividing by n
  - or, a is congruent to b modulo n
- every integer is congruent modulo n to exactly one of the integers in the set of least positive residues: $\{0, 1, 2, \ldots, n-1\}$
- properties of congruence: for a fixed positive integer n and any integers a, b, c, d
  - reflexive: $a \equiv a \ (mod \ n)$
  - symmetric: if $a \equiv b \ (mod \ n)$ then $b \equiv a \ (mod \ n)$
  - transitive: if $a \equiv b \ (mod \ n)$ and $b \equiv c \ (mod \ n)$ then $a \equiv c \ (mod \ n)$
  - addition and multiplication rules: if $a \equiv b \ (mod \ n)$ and $c \equiv d \ (mod \ n)$ then $a \pm c \equiv b \pm d \ (mod \ n)$
  - cancellation rule: if $ac \equiv bc \ (mod \ n)$ and $c \neq 0$ then $a \equiv b \ (mod \ \frac{n}{gcd(c,n)})$, if $gcd(c, n) = 1$ then $a \equiv b \ (mod \ n)$
  - associative and distributive: $(a + b) + c \equiv a + (b + c) \ (mod \ n)$, $(ab)c \equiv a(bc) \ (mod \ n)$, and $a(b + c) \equiv ab + bc \ (mod \ n)$
  - if $a \equiv b \ (mod \ n)$ then $a^r \equiv b^r \ (mod \ n)$, for any integer $r \geq 1$
  - $a \equiv 0 \ (mod \ n)$ if and only if $n|a$
- if m and n are coprime and $a \equiv b \ (mod \ m)$ and $a \equiv b \ (mod \ n)$, then $a \equiv b \ (mod \ mn)$
- the Euler Totient Function of a positive integer n, denoted $\phi(n)$, is the number of positive integer not exceeding n which are relatively prime to n
- for any prime p, $\phi(p) = p - 1$, since all numbers less than p are coprime with it
- if m and n are coprime, then $\phi(m)\phi(n) = \phi(mn)$
- Fermat's Little Theorem: If p is a prime and a is any integer, then $a^p \equiv a \ (mod \ p)$
  - if $gcd(a, p) = 1$ then $a^{p-1} \equiv 1 \ (mod \ p)$
- Euler-Fermat Theorem: if n is a positive integer and a is any integer with $gcd(a, n) = 1$, then $a^{\phi(n)} \equiv 1 \ (mod \ n)$
- the Carmichael function of a positive integer n, denoted $\lambda(n)$, if defined as the smallest positive integer m such that $a^m \equiv 1 \ (mod \ n)$ for all integers relatively prime to n
  - it can be shown that $\lambda(n)$ divides $\phi(n)$
- the least common multiple of the non-zero integers a and b, denoted $lcm(a, b)$, is the smallest positive integer that is a multiple of both a and b
- if n is the product of two distinct primes p and q, $n = pq$, then $\lambda(pq) = lcm(p - 1, q - 1)$
- the modular multiplicative inverse or modular inverse of an integer a modulo n is an integer x such that $ax \equiv 1 \ (mod \ n)$

– write $x = a^{-1}\ mod\ n$ or $x = (\frac{1}{a})\ mod\ n$

– a modular inverse exists if and only if $gcd(a, n) = 1$

5 Linear Programming: An Intro

In general, a linear programmming problem is one that involves the maximization or minimization of a linear function subject to linear constraints. The constraints may be equalities or inequalities.

5.1 Standard Problems

There are two kinds of standard problems, the standard maximum problem and the standard minimum problem. All linear programming problems can be converted into standard form.

The Standard Maximum Problem

Given, an m-vector, $\mathbf{b} = (b_1 \dots b_m)^T$,and n-vector, $\mathbf{c} = (c_1 \dots c_n)^T$, and an $m \times n$ matrix,

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix}$$

of real numbers

The standard maximum problem is: find an n-vector, $\mathbf{x} = (x_1, \dots, x_n)^T$, to maximise:

$$\mathbf{c}^T\mathbf{x} = c_1x_1 + \dots + c_nx_n$$

subject to the constraints:

$$\begin{aligned} &a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n \le b_1 \\ &a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n \le b_2 \\ &\vdots \qquad\qquad\qquad\qquad\qquad\quad \vdots \qquad (\text{or } \mathbf{Ax} \le \mathbf{b}) \\ &a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n \le b_m \end{aligned}$$

and

$$x_1 \ge 0,\ x_2 \ge 0,\ \dots,\ x_m \ge 0 \qquad (\text{or } \mathbf{x} \ge \mathbf{0})$$

The Standard Minimum Problem

Find an m-vector, $\mathbf{y} = (y_1, \dots, y_m)$, to minimize

$$\mathbf{y}^T\mathbf{b} = y_1b_1 + \dots + y_mb_m$$

subject to the constraints:

$$\begin{aligned} &y_1a_{11} + y_2a_{21} + \dots + y_ma_{m1} \ge c_1 \\ &y_1a_{21} + y_2a_{22} + \dots + y_ma_{m2} \ge c_2 \\ &\vdots \qquad\qquad\qquad\qquad\qquad\quad \vdots \qquad (\text{or } \mathbf{y}^T\mathbf{A} \ge \mathbf{c}^T) \\ &y_1a_{1n} + y_2a_{2n} + \dots + y_ma_{mn} \ge c_n \end{aligned}$$

and

$$y_1 \ge 0,\ y_2 \ge 0,\ \dots,\ y_m \ge 0 \qquad (\text{or } \mathbf{y} \ge 0)$$

Converting to Standard Form

- All linear programming problems can be converted to standard form
- a minimum problem can be converted to a maximum by multiplying the objective function by -1
  - constraints of the form: $\sum_{j=1}^{n} a_{ij}x_j \ge b_i$ can be changed into the form: $\sum_{j=1}^{n}(-a_{ij}x_j \ge -b_i$

– from this conversion, two problems arise:

1. Some constraints may be equalities: an equality constraint $\sum_{j=1}^{n} a_{ij}x_j = b_i$ may be removed, by solving this constraint for some x_j for which $a_{ij} \neq 0$ and substituting this solution into the other constraints and into the objetive function wherever x_j appears, this removes one constraint and one variable from the problem
2. Some variable may not be restricted to be non-negative: an unrestricted variable, x_j may be replaced by the difference of two non-negative variables, $x_j = u_j - v_j$, where $u_j \geq 0$ and $v_j \geq 0$, this adds one variable and two non-negative constraints to the problem

5.2 Duality

To every linear program, there is a dual linear program for which it is deeply connected.

Standard Program Duality

Standard program: $\mathbf{c}$ and $\mathbf{x}$ are n-vectors, $\mathbf{b}$ and $\mathbf{y}$ are m-vectors, and $\mathbf{A}$ is an $m \times n$ matrix. Assume $m \geq 1$ and $n \geq 1$

Definition: The dual of the standard maximum problem,

maximize $\mathbf{c}^T\mathbf{x}$

subject to the constraints $\mathbf{Ax} \leq \mathbf{b}$ and $\mathbf{x} \geq 0$

is defined to be the standard minimum problem

minimize $\mathbf{y}^T\mathbf{b}$

subject to the constraints $\mathbf{y}^T\mathbf{A} \geq \mathbf{c}^T$ and $\mathbf{y} \geq 0$

If the standard minimum problem that is the dual of the standard maximum problem is transformed into a standard maximum probmem (by multiplying $\mathbf{A}$, $\mathbf{b}$, and $\mathbf{c}$ by -1), its dual, by definition, is a standard minimum problem which, when transformed into a standard maximum problem, by the same process, is the original standard maximum problem. Thus, the dual of the standard minimum problem is the standard maximum problem.
From this it can also be shown that:

Theorem 1: if $\mathbf{x}$ is feasible for the standard maximum problem and if $\mathbf{y}$ is feasible for its dual, then:

$$\mathbf{c}^T\mathbf{t} \leq \mathbf{y}^T\mathbf{b}$$

Proof:

$$\mathbf{c}^T\mathbf{x} \leq \mathbf{y}^{\mathbf{Ax}} \leq \mathbf{t}^T\mathbf{b}$$

The first inequality comes from $\mathbf{x} \geq \mathbf{0}$ and $\mathbf{c}^T \leq \mathbf{y}^T\mathbf{A}$
The second inequality comes from $\mathbf{y} \geq \mathbf{0}$ and $\mathbf{Ax} \leq \mathbf{b}$

Corollary 1: if a standard problem and its dual are both feasible, then both are bounded feasible
Proof: if $\mathbf{y}$ is feasible for the minimum problem, then Theorem 1 shows that $\mathbf{t}^T\mathbf{b}$ is an upper bound for the values of $\mathbf{c}^T\mathbf{x}$ for $\mathbf{x}$ feasible for the maximum problem. Similarly for the converse

Corollary 2: if there exists feasible $\mathbf{x}^*$ and $\mathbf{y}^*$ for a standard maximum problem and its dual such that $\mathbf{c}^T\mathbf{x}^* = \mathbf{y}^{*T}\mathbf{b}$, then both are optimal for their respective problems
Proof: if $\mathbf{x}$ is any feasible vector for the standard maximum problem, then $\mathbf{c}^T\mathbf{x} \leq \mathbf{y}^{*T}\mathbf{b} = \mathbf{c}^T\mathbf{x}^*$, which shows that $\mathbf{x}^*$

is optimal, a symmetric argument works for $\mathbf{y}^*$

From the theorem and its corollaries, a fundamental theorem is derived.

The Duality Theorem: if a standard linear programming problem is bounded feasible, then so is its dual, their values are equal, and there exists optimal vectors for both problems
There are three possibilities for a linear program. It may be feasible bounded (f.b.), feasible unbounded (f.u.), or infeasible (i). For a program and its dual, there are nine possibilites, three of which are precluded by corollary 1. If a problem and its dual are both feasible, then both must be bounded feasible. Two other possibilites are removed by the Duality Theorem. If a program is feasible bounded, its dual cannot be infeasible.
There are four possibilites for a standard maximum problem and its dual
• both feasible bounded
• standard maximum problem infeasible, its dual feasible unbounded
• standard maximum problem feasible unbounded, its dual infeasible
• both infeasible

From the Duality Theorem, a corollary is found:
The Equilibrium Theorem: let $\mathbf{x}^*$ and $\mathbf{y}^*$ be feasible vectors for a standard maximum problem and its dual, respectively, then $\mathbf{x}^*$ and $\mathbf{y}^*$ are optimal if, and only if:

$$y_i^* = 0 \text{ for all } i \text{ for which } \sum_{j=1}^{n} a_{ij}x_j^* < b_i$$

and

$$x_j^* = 0 \text{ for all } j \text{ for which } \sum_{i=1}^{m} y_i^* a_{ij} > c_j$$

Proof: the first equation in the theorem implies that $y^* = 0$ unless there is equality in $\sum_j a_{ij}x_j^* \leq b_i$

therefore:

$$\sum_{i=1}^{m} y_i^* b_i = \sum_{i=1}^{m} y_i^* \sum_{j=1}^{n} a_{ij}x_j^* = \sum_{i=1}^{m}\sum_{j=1}^{n} y_i^* a_{ij} x_j^*$$

similarly, the following equation implies:

$$\sum_{i=1}^{m}\sum_{j=1}^{n} y_i^* a_{ij} x_j^* = \sum_{j=1}^{n} c_J x_j^*$$

6 Glossary

- **basis**
  - a set of vectors in a vector space, V, is called a basis if the vectors are linearly independent and every vector in V is a linear combination of this set
- **big endian**
  - the most significant byte is stored at an address, and subsquent bytes are stored in lower addresses
- **bit**
  - a binary digit, a 1 or a 0
- **block cipher**
  - a determistic algorithm which operates on a fixed-length group of bits, called blocks, with an unvarying transformation specified by a symmetric key
- **bounded**
  - for a standard maximum or minimum problem, not unbounded
- **byte**
  - a unit of digital information; typically (read: always) 8 bits
  - historically the number of bits used to encode a single character
- **column space**
  - the set of all possible linear combinations of a matrix's column vectors
- **conjugate transpose**
  - the conjugate transpose of mxn matrix A is the matrix, A^* obtained from A by taking the transpose and the taking the complex conjugate of each entry
- **constraint set**
  - the set of feasible vectors
- **coprime**
  - two integers, a and b, whose only common factor is one
  - can also be stated as: their greatest common divisor is one, $gcd(a, b) = 1$
  - also called relatively prime, or written as $(a, b) = 1$, or $a \perp b$
- **deterministic algorithm**
  - an algorithm which, given the same inputs, will always return the same output
- **dictionary coder**
- **digest**
  - the output of a hash function, also called a hash
- **discrete Fourier transform**
  - an algorithm which takes a finite sequence of evenly spaced samples of a function and converts them into the list of coefficients of a finite combination of complex sinusoids, ordered by their frequencies
  - the Fourier analysis of finite-domain (or periodic) discrete-time functions
- **eigenvalue**
  - a scalar value, λ, such that: $T\mathbf{v} = \lambda\mathbf{v}$ is non-zero and holds for some linear transformation T on a vector space V for some non-zero vector $\mathbf{v} \in V$
- **eigenvector**
  - given a linear transformation, T, on a vector space, V, and that $T\mathbf{v} = \lambda\mathbf{v}$ is true and non-zero for some scalar λ(eigenvalue), $\mathbf{v} \in V$ is the eigenvector associated with λ
- **Euler's Totient function**

  - $\phi(n)$, a function which counts the positive integers less than or equal to n that are coprime with n
- **feasible**
  - a vector $\mathbf{x}$, for the standard maximum problem, or $\mathbf{y}$ for the standard minimum problem, is said to be feasible if it satisfies the constraints
  - a linear programming problem is said to be feasible if the constraint set is not empty
- **Fourier analysis**
  - the study of the way functions may be represented or approximated by sums of trigonometric functions
- **Fourier transform**
  - an algorithm which decomposes a function of time into the frequencies that compose it
  - the result is a complex valued function of frequency
    - the absolute value (magnitude) represents the amount of that frequency in the original function
    - the complex argument is the phase shift of the basic sinusoid in that frequency
- **hash**
  - generally, a numerical sum representing some data
- **hash function**
  - a function that maps arbitrarily sized data to data of a fixed size, a hash
- **hexadecimal**
  - the base-16 number system
  - digits are: `0 1 2 3 4 5 6 7 8 9 A B C D E F`
- **identity matrix**
  - the simplest non-trivial diagonal matrix, I such that: $I(X) = X$
- **infeasible**
  - a linear programming problem is said to be infeasible if the constraint set it empty
- **inner product space**
  - a vector space with an additional structure: the inner product which associates each pair of vectors to a scalar, known as the inner product of the vectors
- **integer relation**
- **Krylov subspace**
- **linear programming**
- **linear time**
  - a time complexity, the time to run an algorithm is directly proportional to the length of the input
  - represented in big-O notation as: $O(n)$
- **little endian**
  - the most significant byte is stored at an address, and subsequent bytes are stored in higher addresses
- **logarithmic time**
  - a time complexity, the time to run an algorithm is proportional to the length of the input times the log of the length of the input
  - represented in big-O notation as: $O(nlog(n))$
- **lossless compression**
  - data compression in which the original data can be exactly reconstructed from the compressed data
- **lossy compression**
  - data compression that uses inexact approximations to represent the original data
- **main diagonal**
  - for a matrix, A, composed of elements $a_{i,j}$, the main diagonal is the collection of entries such that: $i = j$

- **matrix decomposition**
  - the factorization of a matrix into the product of matrices
- **matrix multiplication**
  - given $\mathbf{A}$, a $n\times$, matrix and $\mathbf{B}$, a $m \times p$, their product, $\mathbf{AB}$, is an $n \times p$ matrix
  - each entry, of $\mathbf{AB}$ is defined as $(\mathbf{AB})_{ij} = \displaystyle\sum_{k=1}^{m} \mathbf{A}_{ik}\mathbf{B}_{kj}$
  - $\mathbf{AB}_{ij}$ = the ith row of $\mathbf{A}$ dotted with the jth column of $\mathbf{B}$
- **Markov chain**
- **Monte Carlo methods**
- **multipole**
- **n-body problem**
- **objective function**
  - in linear programming, the function to be maximized or minmized
- **octet**
  - a sequence of eight bits, i.e., a byte
- **orthogonal matrix**
  - a matrix, A, is orthogonal if: $A^TA = I$
    - A^T is the transpose of A
    - I is the identity matrix
  - an orthogonal matrix is always invertible and $A^{-1} = A^T$
- **orthonormal basis**
  - a basis for a finite dimensional vector space, V, whose vectors are orthonormal, i.e., they are all orthogonal and unit vectors
- **prefix code**
- **priority queue**
  - a data structure which orders its elements by priority
  - elements with the highest priority are at the front of the queue
- **public-key encryption**
  - cryptographic algorithms which use different keys for encryption and decryption
  - one key is kept private, the other may be published freely
- **queue**
  - a data structure that follow a first-in first-out (FIFO) principle
- **run-length encoding**
- **simplex**
- **sliding window**
- **span**
  - the span of a set of vectors in a vector space is the intersection of all subspaces containing the set
- **symmetric-key encryption**
  - cryptographic algorithms which use the same key for encryption and decryption
- **time complexity**
- **transpose**
  - the transpose of matrix, A is the matrix A^T, it can be created several ways
    - reflecting A over its main diagonal
    - writing the rows of A as the columns of A^T

        - writing the columsn of A as the rows of A^T
        - more fomally, the ith row, jth column element of A^T is the jth row, ith column element of A
            - $[A^T]_{i,j} = [A]_{j,i}$
        - if A is an $m\ x\ n$ matrix, A^T is an nxm matrix

- **unbounded**
    - for maximum problems:
        - the problem is said to be unbounded if the objective function can assume arbitrarily large positive values at feasible vectors
    - for minimum problems:
        - the problem is said to be unbounded if the objective function can assume arbitrarily large negative values at feasible vectors
- **unitary matrix**
    - a matrix, A, whose conjugate transpose A^* is also its inverse, i.e., $A^*A = I$, where I is the identity matrix
- **upper triangular matrix**
    - a matrix in which all the entries below the main diagonal are zero
- **value**
    - for maximum problems:
        - the maximum value of the objective function as the variables range over the constraint set
    - for minimum problems:
        - the minimum value of the objective function as the variables range over the constraint set
- **word**
    - a unit of data for a particular processor