Basic Concepts in Kinematics

Reading material: Chapter 3

Vector of Coordinates

- Vectors of coordinates are designated in this course by column vectors :

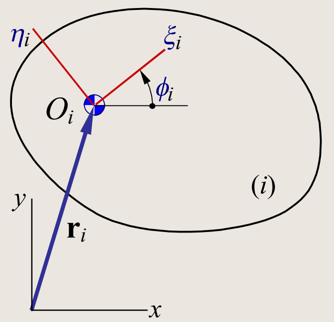
$$\mathbf{q} = \begin{bmatrix} q_1 \\ q_2 \\ \vdots \\ q_n \end{bmatrix}_{n\times 1} = \begin{bmatrix} q_1 & q_2 & \cdots & q_n \end{bmatrix}^T$$

n : the *total number of coordinates* used in describing the system

- Generalized coordinates: a set of variables that completely define the *location* and *orientation* of each body in the system

Global and Body-fixed Coordinate Systems (Planar Motion)

- The vector of coordinates for body *i* in a plane :

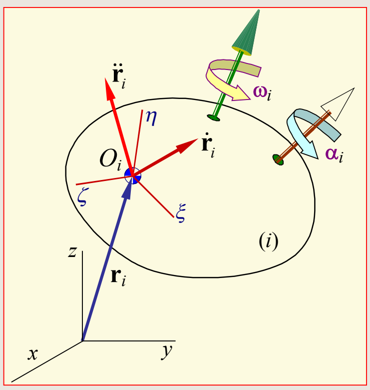


$$\mathbf{q}_i = \begin{bmatrix} x \\ y \\ \phi \end{bmatrix}_i$$

2007/4/4 計算動力學(C.F. Chang) 3

Global and Body-fixed Coordinate Systems (Spatial Motion)

- The vector of coordinates for body *i* in the *three-dimensional space*



$$\mathbf{q}_i = \begin{bmatrix} x \\ y \\ z \\ \phi_1 \\ \phi_2 \\ \phi_3 \end{bmatrix}_i \quad \mathbf{q}_i = \begin{bmatrix} x \\ y \\ z \\ e_0 \\ e_1 \\ e_2 \\ e_3 \end{bmatrix}_i$$

2007/4/4 計算動力學(C.F. Chang) 4

The overall vector of coordinates for a system

- If a mechanism with ***b*** bodies is considered, the number of coordinates required is
- ***n = 3 × b*** if the system is **planar**,
- and
- ***n = 6 × b*** if the system is **spatial**.

$$\mathbf{q} = \begin{bmatrix} \mathbf{q}_1^T & \mathbf{q}_2^T & \cdots & \mathbf{q}_b^T \end{bmatrix}^T$$

For planar system, we have

$$\mathbf{q} = \begin{bmatrix} [x_1 \;\; y_1 \;\; \phi_1] & [x_2 \;\; y_2 \;\; \phi_2] & \cdots & [x_b \;\; y_b \;\; \phi_b] \end{bmatrix}^T$$

The overall vector of coordinates for a system (cont)

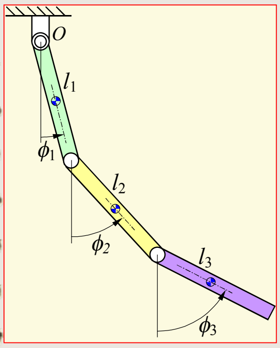
$$\mathbf{q} = \begin{bmatrix} x_1 \\ y_1 \\ \phi_1 \\ x_2 \\ y_2 \\ \phi_2 \\ \vdots \\ x_b \\ y_b \\ \phi_b \end{bmatrix}_{(3b)\times 1}$$

Body *1*

Body *b*

Degrees of Freedom

- **DOF** is the *minimum number* of coordinates required to fully describe the configuration of a system



- **DOF** = 3

because we need at least *three angles* to uniquely determine the configuration of the system

- **DOF** k = n - m

= 3 (num of moving bodies) - num of constraints

= 3(3) - 3(2)

= 9 - 6 = 3

Constraint Equations

- A *kinematic pair* impose certain conditions on the relative motion between the two bodies it comprises.
- When the conditions are expressed in analytical form, they are called *equations of constraint*

Holonomic Constraints (完全拘束)

- **Holonomic constraint:** $\boxed{\mathbf{\Phi}(\mathbf{q},t)=0}$
  - Depend only on *body positions*,
  - Can be Integrated into a form *involving only coordinates* **q**
  - There are two types of holonomic constraints:
    - ***Sclernomic***: *time–independent constraint, i.e.,* **Φ=Φ(q)=0.** *For instance, constraints for perfect joints*
    - ***Rheonomic***: *time–dependent constraint, i.e.,* **Φ=Φ(q,t)= 0.** *For instance,, e.g., constraints for drivers*

$$\boxed{\Phi=\phi-\left(\phi^0+\omega t+\tfrac{1}{2}\alpha t^2\right)=0}$$

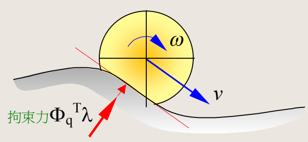
- **Non-holonomic constraint:** $\boxed{\mathbf{\Phi}(\mathbf{q},\dot{\mathbf{q}},t)=0}$
  - Cannot be written in the form of **Φ(q,t) = 0**
  - Involving velocity dq/dt

2007/4/4 計算動力學(C.F. Chang) 9

Non-Holonomic Constraints (不完全拘束)

- One-sided, nonrestrictive constraints

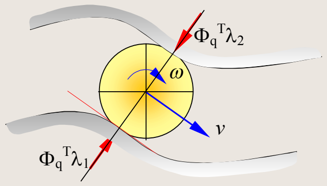
$$\boxed{\mathbf{\Phi}(\mathbf{q},\dot{\mathbf{q}},t)\geq 0}$$



Detach? Contact?
Rolling?
Sliding?
Rolling+Sliding?

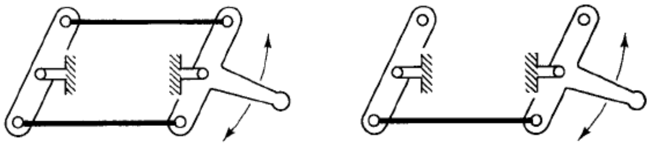
- Two-sided, restrictive constraints

$$\boxed{\mathbf{\Phi}(\mathbf{q},\dot{\mathbf{q}},t)=0}$$



2007/4/4 計算動力學(C.F. Chang) 10

Ex1: Redundant Constraints



k=(4×3)-(6×2)=0 (a) k=(3×3)-(4×2)=1 (b)

Figure 3.8 (a) A double parallel-crank mechanism, and (b) its kinematically equivalent system.

For a system having m *independent* constraint equations and n coordinates, the number of degrees of freedom is determined as follows:

$$\boxed{k = n - m} \tag{3.4}$$

In planar motion, a moving body can have three coordinates, and a revolute joint introduces two constraint equations. For the mechanism of Fig. 3.8(b), there are three moving bodies ($n = 3 \times 3 = 9$) and four revolute joints ($m = 4 \times 2 = 8$). Therefore, $k = 9 - 8 = 1$ DOF. However, for the mechanism of Fig 3.8(a), $n = 4 \times 3 = 12$ and $m = 4 \times 3 = 12$, which yields $k = 12 - 12 = 0$ DOF, which is obviously incorrect. Therefore, Eq. 3.4 yields a correct answer only when the m constraint equations are independent.

Example 2: Redundant Constraints

Five coordinates $\mathbf{q} = [l_1, \phi_1, l_2, \phi_2, l_3]^T$ are used to describe the configuration of bodies in a mechanism. Determine the number of degrees of freedom of the system if the coordinates are dependent according to the following six constraint equations:

$$\Phi_1 = 6\cos\phi_1 - l_2 = 0$$
$$\Phi_2 = 6\sin\phi_1 - l_3 = 0$$
$$\Phi_3 = l_1\cos\phi_1 - 2\cos\phi_2 = 0$$
$$\Phi_4 = l_1\sin\phi_1 - 2\sin\phi_2 - l_3 + 3 = 0$$
$$\Phi_5 = 2\cos\phi_2 + (6 - l_1)\cos\phi_1 - l_2 = 0$$
$$\Phi_6 = 2\sin\phi_2 + (6 - l_1)\sin\phi_1 - 3 = 0$$

Solution An investigation of the six equations reveals that $\Phi_2 = \Phi_4 + \Phi_6$ and $\Phi_1 = \Phi_3 + \Phi_5$.[†] Therefore, two of the equations are redundant, and hence, $m = 6 - 2 = 4$. Since $n = 5$, then $k = 5 - 4 = 1$ DOF.

Redundant Constraints (cont'd)

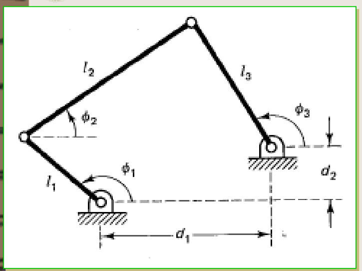
- Question:
  - How to detect redundant constraints automatically?
- Hint:
  - constraint equations are not totally independent if redundant constraints exist (jacobian中至少有一列可由其他列導出)
  - The constraint Jacobian matrix does not have a full row rank ***(rank deficiency***, ***singular!***)
  - Gaussian elimination or LU factorization technique leads to zero rows of Jacobian matrix
  - The number of zero rows is equal to the number of the dependent constraint equations
- Robust and reliable ?
  - We will talk about this later

Kinematic Analysis

- Need only consider constraint equations
- Two methods are introduced in the following:
  - Coordinate Partitioning
  - Appended Driving Constraints
- Velocity equation & acceleration equation:
  - velocity equations: the first time derivatives of the constraint equations
  - acceleration equations: the second time derivatives of the constraint equations

Coordinate Partitioning Method

- Position analysis: determine the configuration of the system
- Given: $\Phi(\mathbf{q}) = 0$, $\Phi_{m\times 1}$, $\mathbf{q}_{n\times 1} = [q_1, q_2, \ldots, q_n]^T$
  - m : the number of independent constraints (equations)
  - n : the number of independent coordinates ($n \geq m$)
- We must specify k (= n - m) driving constraints to solve for the m unknowns



$$
\begin{aligned}
l_1 \cos\phi_1 + l_2 \cos\phi_2 - l_3 \cos\phi_3 - d_1 &= 0 \\
l_1 \sin\phi_1 + l_2 \sin\phi_2 - l_3 \sin\phi_3 - d_2 &= 0
\end{aligned}
$$

- $\mathbf{q}=[\phi_1\ \phi_2\ \phi_3]^T$ = coordinate vector
- m=2, n=3, k=n-m=1
- ϕ_2 and ϕ_3 can be solved by specifying ϕ_1

Coordinate Partitioning Method -Basic concept-

$$
\mathbf{q} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_m \\ v_1 \\ \vdots \\ v_k \end{bmatrix}_{(m+k)\times 1}
$$

- Let $\mathbf{q}=[\mathbf{u}^T, \mathbf{v}^T]^T$
  - **u**: dependent coordinates (須滿足拘束方程式,unknown)
  - **v**: independent coordinates (可任意指定,known)
- Then,
  $\Phi(\mathbf{q}) = 0 \rightarrow \Phi(\mathbf{u}, \mathbf{v}) = 0$
- There are m equations with m unknown **u**
- However, constraint equations are in general ***nonlinear***, that is, they cannot be expressed in the following form:

$$
\mathbf{A}\,\mathbf{u} = \mathbf{b} \quad \not\rightarrow \quad \mathbf{u} = \mathbf{A}^{-1}\,\mathbf{b}
$$

- The configuration of the system can be determined by solving nonlinear equations using iterative numerical method, eg., Newton-Raphson method.

Coordinate Partitioning Method
-Basic concept-

$$\boxed{\boldsymbol{\Phi}=0 \Rightarrow \frac{d\boldsymbol{\Phi}}{dt}=\frac{\partial \boldsymbol{\Phi}}{\partial \mathbf{q}}\frac{d\mathbf{q}}{dt}=\boldsymbol{\Phi}_{\mathbf{q}}\dot{\mathbf{q}}=0}$$

$$d\boldsymbol{\Phi}=\frac{\partial \boldsymbol{\Phi}}{\partial q_1}dq_1+\frac{\partial \boldsymbol{\Phi}}{\partial q_2}dq_2+\ldots+\frac{\partial \boldsymbol{\Phi}}{\partial q_n}dq_n$$

$$=\begin{bmatrix}\frac{\partial \boldsymbol{\Phi}}{\partial q_1} & \frac{\partial \boldsymbol{\Phi}}{\partial q_2} & \cdots & \frac{\partial \boldsymbol{\Phi}}{\partial q_n}\end{bmatrix}_{m\times n}\begin{bmatrix}dq_1\\ dq_2\\ \vdots \\ dq_n\end{bmatrix}_{n\times 1}=\frac{\partial \boldsymbol{\Phi}}{\partial \mathbf{q}}d\mathbf{q}$$

$$\mathbf{q}=\begin{bmatrix}u_1\\ u_2\\ \vdots\\ u_m\\ v_1\\ \vdots\\ v_k\end{bmatrix}_{(3b)\times 1}$$

Coordinate Partitioning Method
-Velocity Analysis-

$$\boldsymbol{\Phi}_{\mathbf{q}}\dot{\mathbf{q}}=\left[\begin{array}{ccc|ccc}\frac{\partial \Phi_1}{\partial u_1} & \cdots & \frac{\partial \Phi_1}{\partial u_m} & \frac{\partial \Phi_1}{\partial v_1} & \cdots & \frac{\partial \Phi_1}{\partial v_k}\\ \vdots & & & & & \vdots\\ \frac{\partial \Phi_m}{\partial u_1} & \cdots & \frac{\partial \Phi_m}{\partial u_m} & \frac{\partial \Phi_m}{\partial v_1} & \cdots & \frac{\partial \Phi_m}{\partial v_k}\end{array}\right]_{m\times n}\begin{bmatrix}\dot{u}_1\\ \vdots\\ \dot{u}_m\\ \hline \dot{v}_1\\ \vdots\\ \dot{v}_k\end{bmatrix}_{n\times 1}$$

$$\dot{\mathbf{q}}=\begin{bmatrix}\dot{u}_1\\ \vdots\\ \dot{u}_m\\ \dot{v}_1\\ \vdots\\ \dot{v}_k\end{bmatrix}_{n\times 1}$$

$$=[\boldsymbol{\Phi}_{\mathbf{u}}\boldsymbol{\Phi}_{\mathbf{v}}]\begin{bmatrix}\dot{\mathbf{u}}\\ \dot{\mathbf{v}}\end{bmatrix}=\underline{\boldsymbol{\Phi}_{\mathbf{u}}\dot{\mathbf{u}}+\boldsymbol{\Phi}_{\mathbf{v}}\dot{\mathbf{v}}=0}$$

$$\boxed{\boldsymbol{\Phi}_{\mathbf{u}}\dot{\mathbf{u}}=-\boldsymbol{\Phi}_{\mathbf{v}}\dot{\mathbf{v}}}$$

$$\mathbf{A}_{m\times m}\,\mathbf{x}_{m\times 1}=\mathbf{b}_{m\times 1}$$

Dependent velocities $\dot{\mathbf{u}}$ can be solved by using linear algebra

Coordinate Partitioning Method
-Acceleration Analysis-

$$\boxed{\Phi = \Phi(\mathrm{q})} \qquad \ddot{\mathbf{q}} = \begin{bmatrix} \ddot{u}_1 \\ \vdots \\ \ddot{u}_m \\ \ddot{v}_1 \\ \vdots \\ \ddot{v}_k \end{bmatrix}_{n\times 1}$$

$$\dot{\mathbf{\Phi}} = \frac{d\mathbf{\Phi}}{dt} = \frac{\partial \mathbf{\Phi}}{\partial \mathbf{q}}\frac{d\mathbf{q}}{dt} = \mathbf{\Phi}_{\mathrm{q}}\dot{\mathbf{q}} = 0$$

$$\ddot{\mathbf{\Phi}} = \frac{d}{dt}(\mathbf{\Phi}_{\mathrm{q}}\dot{\mathbf{q}}) = \frac{\partial}{\partial \mathbf{q}}(\mathbf{\Phi}_{\mathrm{q}}\dot{\mathbf{q}})\frac{d\mathbf{q}}{dt} + \frac{\partial}{\partial t}(\mathbf{\Phi}_{\mathrm{q}}\dot{\mathbf{q}})\frac{dt}{dt}$$

$$= (\mathbf{\Phi}_{\mathrm{q}}\dot{\mathbf{q}})_{\mathrm{q}}\dot{\mathbf{q}} + (\underbrace{\mathbf{\Phi}_{\mathrm{q}t}}_{0}\dot{\mathbf{q}} + \mathbf{\Phi}_{\mathrm{q}}\ddot{\mathbf{q}}) = \boxed{(\mathbf{\Phi}_{\mathrm{q}}\dot{\mathbf{q}})_{\mathrm{q}}\dot{\mathbf{q}} + \mathbf{\Phi}_{\mathrm{q}}\ddot{\mathbf{q}} = 0}$$

$$\mathbf{\Phi}_{\mathrm{q}}\ddot{\mathbf{q}} = [\mathbf{\Phi}_{\mathrm{u}}\mathbf{\Phi}_{\mathrm{v}}]\begin{bmatrix} \ddot{\mathbf{u}} \\ \ddot{\mathbf{v}} \end{bmatrix} = \mathbf{\Phi}_{\mathrm{u}}\ddot{\mathbf{u}} + \mathbf{\Phi}_{\mathrm{v}}\ddot{\mathbf{v}} = -(\mathbf{\Phi}_{\mathrm{q}}\dot{\mathbf{q}})_{\mathrm{q}}\dot{\mathbf{q}}$$

$$\boxed{\mathbf{\Phi}_{\mathrm{u}}\ddot{\mathbf{u}} = -\mathbf{\Phi}_{\mathrm{v}}\ddot{\mathbf{v}} - (\mathbf{\Phi}_{\mathrm{q}}\dot{\mathbf{q}})_{\mathrm{q}}\dot{\mathbf{q}}}$$

$$\mathbf{A}_{m\times m}\,\mathbf{x}_{m\times 1} = \mathbf{b}_{m\times 1}$$

Dependent accelerations $\boxed{\ddot{\mathbf{u}}}$ can be solved if $\boxed{\ddot{\mathbf{v}}}$ and $\boxed{\dot{\mathbf{q}}}$ are known

2007/4/4 計算動力學(C.F. Chang) 19

Method of Appended Driving Constraints
-Position Analysis-

- Providing that there are *m* constraints
- When appending *k* driving constraints to the system, it has $n = m + k$ equations in n variables:

$\mathbf{\Phi} = \mathbf{\Phi}(\mathbf{q}) = 0$	*m* kinematic constraints
$\mathbf{\Phi}^{(d)} = \mathbf{\Phi}(\mathbf{q},t) = 0$	*k* driving constraints

- In general, these equations are also nonlinear
- So, the configuration of the system may be determined by using iterative numerical method

2007/4/4 計算動力學(C.F. Chang) 20

Method of Appended Driving Constraints
-Velocity Analysis-

For kinematic constraints : $\boxed{\Phi = \Phi(\mathbf{q})}$

$$\boxed{\dot{\mathbf{\Phi}} = \mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}} = 0}$$

For driving constraints: $\boxed{\Phi = \Phi(\mathbf{q}\,,\, t)}$

$$\boxed{\dot{\mathbf{\Phi}}^{(d)} = \mathbf{\Phi}_{\mathbf{q}}^{(d)}\dot{\mathbf{q}} + \mathbf{\Phi}_{t}^{(d)} = \mathbf{0}}$$

$$\underbrace{\begin{bmatrix} \mathbf{\Phi}_{\mathbf{q}} \\ \mathbf{\Phi}_{\mathbf{q}}^{(d)} \end{bmatrix}}_{\mathbf{A}_{m\times m}} \underbrace{\dot{\mathbf{q}}}_{\mathbf{x}_{m\times 1}} = \underbrace{\begin{bmatrix} 0 \\ -\mathbf{\Phi}_{t}^{(d)} \end{bmatrix}}_{\mathbf{b}_{m\times 1}}$$

$\mathbf{A}_{m\times m}\ \mathbf{x}_{m\times 1} = \mathbf{b}_{m\times 1}$

velocities can be determined by solving Ax=b

Method of Appended Driving Constraints
-Acceleration Analysis-

For kinematic constraints : $\boxed{\Phi = \Phi(\mathbf{q})}$ $\boxed{\dot{\mathbf{\Phi}} = \mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}} = 0}$

$$\boxed{\ddot{\mathbf{\Phi}} = (\mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}\ddot{\mathbf{q}} = 0}$$

For driving constraints: $\boxed{\Phi = \Phi(\mathbf{q}\,,\, t)}$ $\boxed{\dot{\mathbf{\Phi}}^{(d)} = \mathbf{\Phi}_{\mathbf{q}}^{(d)}\dot{\mathbf{q}} + \mathbf{\Phi}_{t}^{(d)} = \mathbf{0}}$

$$\begin{aligned}
\ddot{\mathbf{\Phi}}^{(d)} &= \frac{d}{dt}\dot{\mathbf{\Phi}}_{\mathbf{q}}^{(d)} = \frac{d}{dt}(\mathbf{\Phi}_{\mathbf{q}}^{(d)}\dot{\mathbf{q}} + \mathbf{\Phi}_{t}^{(d)}) \\
&= (\mathbf{\Phi}_{\mathbf{q}}^{(d)}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} + (\mathbf{\Phi}_{\mathbf{q}}^{(d)}\dot{\mathbf{q}})_{t} + \mathbf{\Phi}_{tq}^{(d)}\dot{\mathbf{q}} + \mathbf{\Phi}_{tt}^{(d)} \\
&= (\mathbf{\Phi}_{\mathbf{q}}^{(d)}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}t}^{(d)}\dot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}^{(d)}\ddot{\mathbf{q}} + \mathbf{\Phi}_{tq}^{(d)}\dot{\mathbf{q}} + \mathbf{\Phi}_{tt}^{(d)} = 0
\end{aligned}$$

$$\boxed{\therefore (\mathbf{\Phi}_{\mathbf{q}}^{(d)}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} + 2\mathbf{\Phi}_{\mathbf{q}t}^{(d)}\dot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}^{(d)}\ddot{\mathbf{q}} + \mathbf{\Phi}_{tt}^{(d)} = 0}$$

Method of Appended Driving Constraints -Acceleration Analysis (cont'd)

$$(\mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}\ddot{\mathbf{q}} = 0$$

Kinematic constraints

$$(\mathbf{\Phi}_{\mathbf{q}}^{(d)}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} + 2\mathbf{\Phi}_{\mathbf{q}t}^{(d)}\dot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}^{(d)}\ddot{\mathbf{q}} + \mathbf{\Phi}_{tt}^{(d)} = 0$$

Driving constraints

$$\begin{bmatrix} \mathbf{\Phi}_{\mathbf{q}} \\ \mathbf{\Phi}_{\mathbf{q}}^{(d)} \end{bmatrix} \ddot{\mathbf{q}} = \begin{bmatrix} -(\mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} \\ -(\mathbf{\Phi}_{\mathbf{q}}^{(d)}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} - 2\mathbf{\Phi}_{\mathbf{q}t}^{(d)}\dot{\mathbf{q}} - \mathbf{\Phi}_{tt}^{(d)} \end{bmatrix}$$

$$\mathbf{A}_{m\times m}\,\mathbf{x}_{m\times 1} = \mathbf{b}_{m\times 1}$$

accelerations can be determined by solving Ax=b

Newton-Raphson method
One Equation in One Unknown

- Considering an nonlinear equation $\Phi(x)=0$, we have

$$\Phi_x \equiv \frac{d\Phi}{dx} \cong \frac{\Phi(x^{j+1}) - \Phi(x^j)}{x^{j+1} - x^j}$$

- In order that $\Phi(x^{j+1})=0$, we must specify x^{j+1} to satisfy

$$\Phi_x = \frac{0 - \Phi(x^j)}{x^{j+1} - x^j}$$

- Thus, we will approach a root of $\Phi(x^{j+1})=0$ by selecting

$$x^{j+1} = x^j - \frac{\Phi(x^j)}{\Phi_x}$$

Physical Interpretation
for Newton-Raphson method

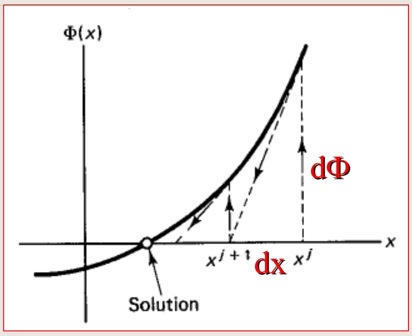
$$\Phi_x \equiv \frac{d\Phi}{dx} \cong \frac{\Phi(x^{j+1})-\Phi(x^j)}{x^{j+1}-x^j}$$

- If x_1^{j+1} are the root of Φ=0, we have

$$\Phi_x = \frac{0-\Phi(x^j)}{x^{j+1}-x^j}$$

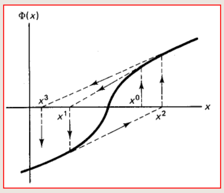
- Then,

$$x^{j+1} = x^j - \frac{\Phi(x^j)}{\Phi_x}$$



$\Phi_x \cong \Delta\Phi/\Delta x=(0-\Phi(x^j))/(x^{j+1}-x^j)$

- A case of failure



Newton-Raphson method
One Equation in One Unknown

```
function test
clc
x0=10
newton1(x0)

%-----------------------------
function y=newton1(x0)
x1=x0-fun(x0)/fq(x0);
n=1;
while (norm(x1-x0)>=1.0e-6)&(n<=1000)
    x0=x1;
    x1=x0-fun(x0)/fq(x0);
    n=n+1;
end
x1 % print solution
n % print num of iteration
```

```
%-----------------------------
function y=fun(x)
y = x*x - 4.0;

%-----------------------------
function y=fq(x)
y=2*x;
```

$$x^{j+1} = x^j - \frac{\Phi(x^j)}{\Phi_x}$$

Newton-Raphson method
2 Equations in 2 Unknowns

$$\begin{aligned} x_1^{j+1} &= x_1^j + \Delta x_1 \\ x_2^{j+1} &= x_2^j + \Delta x_2 \end{aligned} \Rightarrow \boxed{\mathbf{x}^{j+1} = \mathbf{x}^j + \Delta\mathbf{x}}$$

- Consider two nonlinear equations and Taylor theorem

$$\begin{aligned} \Phi_1(x_1, x_2) &= 0 \\ \Phi_2(x_1, x_2) &= 0 \end{aligned} \quad \text{Eq. (1)}$$

$$\Phi_1(x_1^{j+1}, x_2^{j+1}) = \Phi_1(x_1^j, x_2^j) + \Delta x_1 \frac{\partial \Phi_1}{\partial x_1} + \Delta x_2 \frac{\partial \Phi_1}{\partial x_2} + \cdots$$

$$\Phi_2(x_1^{j+1}, x_2^{j+1}) = \Phi_2(x_1^j, x_2^j) + \Delta x_1 \frac{\partial \Phi_2}{\partial x_1} + \Delta x_2 \frac{\partial \Phi_2}{\partial x_2} + \cdots$$

- If (x_1^{j+1}, x_2^{j+1}) are the roots Eq. (1), we get

$$\begin{aligned} 0 &= \Phi_1(x_1^j, x_2^j) + \Delta x_1 \frac{\partial \Phi_1}{\partial x_1} + \Delta x_2 \frac{\partial \Phi_1}{\partial x_2} \\ 0 &= \Phi_2(x_1^j, x_2^j) + \Delta x_1 \frac{\partial \Phi_2}{\partial x_1} + \Delta x_2 \frac{\partial \Phi_2}{\partial x_2} \end{aligned} \Rightarrow \begin{bmatrix} \frac{\partial \Phi_1}{\partial x_1} & \frac{\partial \Phi_1}{\partial x_2} \\ \frac{\partial \Phi_2}{\partial x_1} & \frac{\partial \Phi_2}{\partial x_2} \end{bmatrix} \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \end{bmatrix} = -\begin{bmatrix} \Phi_1 \\ \Phi_2 \end{bmatrix}$$

$$\boldsymbol{\Phi}_{\mathbf{x}}(\mathbf{x}^j)\Delta\mathbf{x} = -\boldsymbol{\Phi}(\mathbf{x}^j)$$

Newton-Raphson method
N Equations in N Unknown

- Similarly, for a set of *n* nonlinear equations with *n* unknowns, we can determine $\Delta\mathbf{x}$ by solving

$$\boldsymbol{\Phi}_{\mathbf{x}}(\mathbf{x}^j)\Delta\mathbf{x} = -\boldsymbol{\Phi}(\mathbf{x}^j)$$

$$\boxed{\mathbf{A}_{n\times n}\,\mathbf{x}_{n\times 1} = \mathbf{b}_{n\times 1}}$$

- Then, $\mathbf{x}^{j+1}$ can be evaluated from

$$\boxed{\mathbf{x}^{j+1} = \mathbf{x}^j + \Delta\mathbf{x}}$$

Example 3.13, p.69

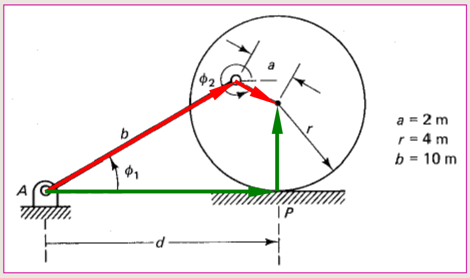
Apply the Newton-Raphson method to find Φ_2 and d when $\Phi_1 = 30°$.

$$\mathbf{b}+\mathbf{a}-\mathbf{r}-\mathbf{d}=\mathbf{0}$$

$$\Phi_1 \equiv b\cos\phi_1 + a\cos\phi_2 - d = 0$$
$$\Phi_2 \equiv b\sin\phi_1 + a\sin\phi_2 - r = 0$$

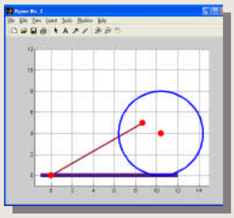
$$\mathbf{q} = [\phi_2, d, \phi_1]^T$$

$$\mathbf{\Phi_q} \equiv \left[\begin{array}{cc|c} -2\sin\phi_2 & -1 & -10\sin\phi_1 \\ 2\cos\phi_2 & 0 & 10\cos\phi_1 \end{array}\right]$$



Iteration number	$\phi_2^\dagger$	d
1	5.59 (326°)	10.0
2	5.77	10.43
3	5.76	10.3924

Initial guess x0



Continuation method (連續法)

Morgan, A. P., Solving Polynomial Systems Using Continuation for Engineering and Scientific Problems, Prentice-Hall, 1987.

- Problem: determine **x** for F(**x**)=0
- Procedure:
  - Define a function G(**x**) having the same degrees as F=0. For instance, Let $G_j = c_{1j}x_j^{dj} - c_{2j} = 0$, where d_j is the degree of F_j=0. (令F和G有相同次數之目的, 在使兩者有相同數目的解)
  - Define Homotopy function(同倫函數) H(X,t) = tF+(1-t)G, t=0~1
  - For equation H=0, we have

$$\dot{\mathbf{H}} = \mathbf{H}_x\dot{\mathbf{x}} + \mathbf{H}_t = \mathbf{0}$$

  - Then, x(t) can be determined by solving the following initial value problem:

$$\mathbf{H}_x\dot{\mathbf{x}} = -\mathbf{H}_t,\ \ \mathbf{x}|_{t=0} = \mathbf{x}(0) = \text{initial value} \qquad \mathbf{x}(t+dt) = \mathbf{x}(t) + \dot{\mathbf{x}}dt$$

- Note that, H=G when t=0. Thus, x(0) is easily determined from eq. G=0 (方程式G是自訂的, 故可定義一個容易求解的方程式).
- The desired x for F=0 is the solution of H when t=1. That is, x(1) is the desired solution.

The Solution of G=0 and De Moivre's Theorem

For $G_j = c_{1j} x_j^{d_j} - c_{2j} = 0$ $(j=1, 2, \ldots, n)$

where c_1 *and* c_2 *are random complex numbers*

De Moivre's Theorem

$$z^n = (re^{i\theta})^n = [r(\cos\theta + i\sin\theta)]^n = r^n e^{in\theta} = r^n(\cos n\theta + i\sin n\theta)$$

we have

- $x_j^{d_j} = c_{2j}/c_{1j} = a + ib = r_j e^{i(\theta_j + 2k\pi)}$, $\quad j=1, 2, \ldots, n$
  - *where* $r_j = (a^2+b^2)^{1/2}$, $\theta_j = \tan^{-1}(b/a)$
- ⇨ $x_j = [r_j e^{i(\theta_j + 2k\pi)}]^{1/d_j}$, $\quad k=0, 2, \ldots, d_j$

$$x_j = (r_j)^{\frac{1}{d_j}} \left[\cos\left(\frac{\theta_j + 2k\pi}{d_j}\right) + i\sin\left(\frac{\theta_j + 2k\pi}{d_j}\right) \right]$$

Summy

$$\dot{\mathbf{\Phi}}^{(d)} = \mathbf{\Phi}_{\mathbf{q}}^{(d)}\dot{\mathbf{q}} + \mathbf{\Phi}_t^{(d)} = \mathbf{0}$$

$$(\mathbf{\Phi}_{\mathbf{q}}^{(d)}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} + 2\mathbf{\Phi}_{\mathbf{q}t}^{(d)}\dot{\mathbf{q}} + \mathbf{\Phi}_{\mathbf{q}}^{(d)}\ddot{\mathbf{q}} + \mathbf{\Phi}_{tt}^{(d)} = 0$$

- Generalized coordinates
  - $\mathbf{q}_{n\times1} = [q_1, q_2, \ldots, q_n]^T$
- Constraints:
  - $\mathbf{\Phi}_{m\times1} = \mathbf{\Phi}(\mathbf{q}, t) = [\Phi_1(\mathbf{q}, t), \Phi_2(\mathbf{q}, t), \ldots, \Phi_m(\mathbf{q}, t),]^T = 0$
- Velocity equations :

$$\mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}} = -\mathbf{\Phi}_t$$

- Acceleration equations :

$$\mathbf{\Phi}_{\mathbf{q}}\ddot{\mathbf{q}} = -[(\mathbf{\Phi}_{\mathbf{q}}\dot{\mathbf{q}})_{\mathbf{q}}\dot{\mathbf{q}} + 2\mathbf{\Phi}_{\mathbf{q}t}\dot{\mathbf{q}} + \mathbf{\Phi}_{tt}] \equiv \boldsymbol{\gamma}$$

The rhs of acceleration equation

Velocity equation and acceleration equation have the same coefficient matrix $\Phi_{\mathbf{q}}$

Question

- In the previous discussion, constraint equations are determined manually.
- Can the constraint equations be generated automatically for general systems?
  - The answer is positive.
  - We'll see how to do that in the next chapter .

Remark on the redundant constraints

$$\underbrace{\mathbf{\Phi}_{\mathbf{x}}(\mathbf{x}^j)}_{\mathbf{A}_{n\times n}}\underbrace{\Delta\mathbf{x}}_{\mathbf{x}_{n\times 1}} = \underbrace{-\mathbf{\Phi}(\mathbf{x}^j)}_{\mathbf{b}_{n\times 1}}$$

$$\mathbf{\Phi}_{\mathbf{q}}\Delta\mathbf{q} = -\mathbf{\Phi}$$

- This equation can be solved by linear solver if
  - m=n, number of constraint equations=number of variables
  - constraint equations are independent
  - no redundant constraints
  - Φ_q is a square matrix and nonsingular
- A reliable algorithm to solve the redundant system of linear equations is the least-square formulation (strang, 1980):

$$(\underbrace{\mathbf{\Phi}_{\mathbf{q}}^T}_{(n\times m)}\underbrace{\mathbf{\Phi}_{\mathbf{q}}}_{(m\times n)})\underbrace{\Delta\mathbf{q}}_{n\times 1} = -\underbrace{\mathbf{\Phi}_{\mathbf{q}}^T}_{(n\times m)}\underbrace{\mathbf{\Phi}}_{(m\times 1)}$$

$$\boxed{\mathbf{A}_{n\times n}\,\mathbf{x}_{n\times 1} = \mathbf{b}_{n\times 1}}$$

Why?

The Least-Square Method for Solving $A_{m\times n}x_{n\times 1}=b_{m\times 1}$, m>n

- 物理意義:
  - (a) 方程式之數目m大於變數之數目n; (➔當方程式相依時才有解)
  - (b) 從小維數之空間映至較大維數之空間, 例如: 以兩個actuator控制End-Effector使滿足指定之三維速度, 即 $[v_x, v_y, v_z]^T = J\,[dq_1/dt, dq_2/dt]^T$
- Concept:
  - determine the $\mathbf{x}(=\Delta\mathbf{q})$ that minimize $\Phi_q\mathbf{x}+\Phi$ for a given Φ and Φ_q.
- Let $\Phi_q\mathbf{x}+\Phi = \mathbf{e}$ =error, and $z = ||\mathbf{e}||^2/2$.
- In order to minimize the error **e**, we need to determine the **x** that satisfies $dz/d\mathbf{x}=0$.

$$\underset{(m\times n)}{\boldsymbol{\Phi}_{\mathbf{q}}}\,\underset{(n\times 1)}{\Delta\mathbf{q}} = \underset{(m\times 1)}{-\boldsymbol{\Phi}}$$

$$\dfrac{dz}{d\mathbf{x}} = \left(\dfrac{d\mathbf{e}}{d\mathbf{x}}\right)^T\left(\dfrac{dz}{d\mathbf{e}}\right) \quad \boxed{\text{Chain rule}}$$
$$= (\Phi_q)^T(\mathbf{e})$$
$$= (\Phi_q)^T(\Phi_q\mathbf{x}+\Phi)$$
$$= \Phi_q^T\Phi_q\mathbf{x}+\Phi_q^T\Phi = 0$$

$$(\underset{(n\times m)}{\boldsymbol{\Phi}_{\mathbf{q}}^T}\,\underset{(m\times n)}{\boldsymbol{\Phi}_{\mathbf{q}}})\,\underset{(n\times 1)}{\Delta\mathbf{q}} = -\underset{(n\times m)}{\boldsymbol{\Phi}_{\mathbf{q}}^T}\,\underset{(m\times 1)}{\boldsymbol{\Phi}}$$

➔ $(\Phi_q^T\Phi_q)\,\mathbf{x} = -\Phi_q^T\Phi$ ← Q.E.D.

$$\boldsymbol{\Phi}_{\mathbf{q}}\Delta\mathbf{q} = -\boldsymbol{\Phi}$$

Pseudo-Inverse of Jacobian Matrix for m≠n

- Concept: determine the minimum of $x=\Delta q$ that satisfies the $\Phi_q x=-\Phi$ for a given Φ and Φ_q.
- Formulating as an optimization problem:
  - Min $F= x^TWx$, W is an symmetric positive definite weighting matrix
  - Subject to equality constraint $h= \Phi_q x+\Phi=0$ (驗證一矩陣是否爲positive definite之最保險的方法爲判斷其特徵值是否皆爲正值)
- Solve this problem using Largrange multipliers. (參見最佳化方法之講義)
  - Let $L= x^TWx-\lambda^T(\Phi_q x+\Phi)$

Then, the x and λ can be determined by solving

$$\frac{\partial L}{\partial \mathbf{x}} = 0 \rightarrow 2\mathbf{W}\mathbf{x} - \boldsymbol{\Phi}_q^T\boldsymbol{\lambda} = 0 \rightarrow \mathbf{x} = \frac{1}{2}\mathbf{W}^{-1}\boldsymbol{\Phi}_q^T\boldsymbol{\lambda}$$

$$\frac{\partial L}{\partial \lambda} = 0 \rightarrow \Phi_q\mathbf{x} + \Phi = 0$$

$$\rightarrow \Phi_q\mathbf{W}^{-1}\boldsymbol{\Phi}_q^T\boldsymbol{\lambda} = -2\Phi \rightarrow \boldsymbol{\lambda} = -2(\Phi_q\mathbf{W}^{-1}\boldsymbol{\Phi}_q^T)^{-1}\Phi$$

$$\mathbf{x} = -\mathbf{W}^{-1}\boldsymbol{\Phi}_q^T(\Phi_q\mathbf{W}^{-1}\boldsymbol{\Phi}_q^T)^{-1}\Phi = -\boldsymbol{\Phi}_q^T(\Phi_q\boldsymbol{\Phi}_q^T)^{-1}\Phi = -\boldsymbol{\Phi}_q^{\#}\boldsymbol{\Phi}$$

Let **W=I**

Pseudo-Inverse of Jacobian Matrix

Remark on the redundant constraints (cont'd)

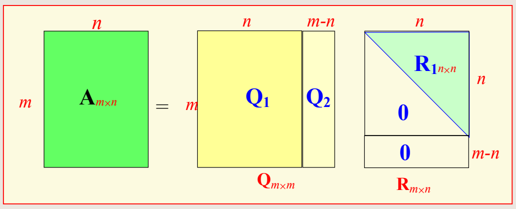
- Methods based on the Q-R or SVD (singular value decomposition) are also available for dealing with those systems with redundant constraints.
  - In general, QR is numerically more efficient, but SVD is more stable in poor numerically conditioned problems.
- QR Decomposition:
  - $\mathbf{A}_{(m\times n)}\mathbf{x}_{(n\times 1)}=\mathbf{b}_{(m\times 1)}$,
    - **A**=**QR**, **Q**:orthogonal matrix, **R**: upper-triangular matrix.
    - This factored form of **A** can be verified by using the concept of Gram-Schmidt orthogonalization (ref: Shabana).
  - If m=n, **Ax=b** ➔ **(QR)x=b** ➔ **Rx** = **Q**$^{-1}$**b** = **Q**^{T}**b**
  - Thus, x can be easily determined by back-substitution because R is a upper-triangular matrix
  - Matlab command: x=A\b. (QR factorization is used to solve linear systems with more equations than unknowns. See MATLAB Function Reference)

Matlab example

Remark on the redundant constraints (cont'd)

- If *m*≠*n*, say *m*>*n* (方程式之數目*m*>變數之數目*n*), one can write

$$\mathbf{A}_{m\times n}=\mathbf{Q}_{m\times m}\mathbf{R}_{m\times n}=\begin{bmatrix}\mathbf{Q}_1 & \mathbf{Q}_2\end{bmatrix}_{m\times m}\begin{bmatrix}\mathbf{R}_1\\ \mathbf{0}\end{bmatrix}_{m\times n}=\mathbf{Q}_1\mathbf{R}_1$$



Ax=b

➔$(\mathbf{Q}_1\mathbf{R}_1)\mathbf{x}=\mathbf{b}_1$

➔ $\mathbf{R}_1\mathbf{x} = \mathbf{Q}_1^{-1}\mathbf{b}_1 = \mathbf{Q}_1^{T}\mathbf{b}_1$

Thus, **x** can also be determined by back-substitution because $\mathbf{R}_1$ is a upper-triangular matrix

Remark on the redundant constraints (cont'd)

- SVD: $\mathbf{Ax=b} \Rightarrow (\mathbf{USV^T})\mathbf{x=b}$
  - $\mathbf{A}_{mxn} = \mathbf{USV^T} \Leftrightarrow \mathbf{A}_{mxn} = \mathbf{U_d D V^T}$
    - $\mathbf{U}$: mxm orthonormal matrix ➔ $\mathbf{U^TU=I}$
    - $\mathbf{S}$: mxn diagonal matrix with $S_{ij}=0$ if $i \neq j$, $S_{ii}>0$ if $I=j$.
    - $\mathbf{V}$: nxn orthonormal matrix ➔ $\mathbf{V^TV=I}$

$\mathbf{A}_{m\times n}\mathbf{x}_{n\times 1} = \mathbf{b}_{n\times 1}$

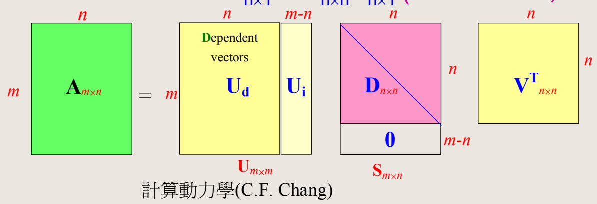
➔ $(\mathbf{USV^T})_{m\times n}\mathbf{x}_{n\times 1} = \mathbf{b}_{n\times 1}$

➔ $\mathbf{S}_{m\times n}\underline{\mathbf{V^Tx}} = \underline{\mathbf{U^Tb}}$ (premultiply both sides by $\mathbf{U}^T$, note: $\mathbf{U^TU=I}$)

➔ $\mathbf{S}_{m\times n}\mathbf{z}_{n\times 1} = \mathbf{d}_{m\times 1}$ (Let $\mathbf{V^Tx=z}_{n\times 1}$, and $\mathbf{U^Tb} = \mathbf{d}_{m\times 1}$)

- $\mathbf{z}_{n\times 1}$ can be easily determined since $\mathbf{S}$ is a diagonal matrix
- Then, $\mathbf{x}$ is determined from $\mathbf{x}_{n\times 1} = \mathbf{V}_{n\times n}\mathbf{z}_{n\times 1}$ (note: $\mathbf{V^Tx=z}$, $\mathbf{V^TV=I}$)

Matlab example



Example: SVD

- Find the singular value decomposition of the matrix $\mathbf{A}_{3\times 2} = \begin{bmatrix} 2 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix}$

Step1:
determine the eigenvalues and eigenvectors of the matrix $\mathbf{A^TA}$

$$\mathbf{A}^T\mathbf{A} = \begin{bmatrix} 2 & 1 & 1 \\ 1 & 0 & 0 \end{bmatrix}\begin{bmatrix} 2 & 1 \\ 1 & 0 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 6 & 2 \\ 2 & 1 \end{bmatrix}$$

From which the eigenvalues can be determined as

$$\lambda_1=0.2984,\ \lambda_2=6.7016$$

The eigenvectors are

$$\mathbf{V} = [\mathbf{x}_1 \quad \mathbf{x}_2] = \begin{bmatrix} 0.3310 & -0.9436 \\ -0.9436 & -0.3310 \end{bmatrix}$$

Matlab command: [V,lamda]=eig(X)

V=

0.3310 -0.9436

-0.9436 -0.3310

lamda=

0.2984 0

0 6.7016

Example: SVD (cont'd)

Step2: determine **U** and **S**

$$\mathbf{AV}=\begin{bmatrix}2&1\\1&0\\1&0\end{bmatrix}\begin{bmatrix}0.3310&-0.9436\\-0.9436&-0.3310\end{bmatrix}=\begin{bmatrix}-0.2816&-2.2183\\0.3310&-0.9436\\0.3310&-0.9436\end{bmatrix}=\begin{bmatrix}\mathbf{b}_1&\mathbf{b}_2\end{bmatrix}$$

$$\mathbf{U}=\begin{bmatrix}\hat{\mathbf{b}}_1&\hat{\mathbf{b}}_2&\hat{\mathbf{b}}_3\end{bmatrix}=\begin{bmatrix}-0.5155&-0.8569&0\\0.6059&-0.3645&-0.7071\\0.6059&-0.3645&0.7071\end{bmatrix}$$

Choose b_3 such that **U** is an orthonormal matrix. Here, we choose $\mathbf{b}_3=\mathbf{b}_1\times\mathbf{b}_2$

$$\hat{\mathbf{b}}_i\equiv\frac{\mathbf{b}_i}{\|\mathbf{b}_i\|}$$

$$\mathbf{S}_{3\times2}=\begin{bmatrix}\sqrt{\lambda_1}&0\\0&\sqrt{\lambda_1}\\0&0\end{bmatrix}=\begin{bmatrix}0.5463&0\\0&2.5887\\0&0\end{bmatrix}$$

$$\mathbf{V}^T=\begin{bmatrix}0.3310&-0.9436\\-0.9436&-0.3310\end{bmatrix}^T=\begin{bmatrix}0.3310&-0.9436\\-0.9436&-0.3310\end{bmatrix}$$

Matlab

Example_svd.m

Implementation with matlab

- clc;
- A = [2 1; 1 0; 1 0]
- disp('Determine the SVD of matrix A step by step:')
- AtA = A'*A
- disp('Eigen vectors V and eigen values lamda of matrix AtA')
- [V,lamda]=eig(AtA)
- % Determine S
- S=zeros(3,2);
- S(1,1)=sqrt(lamda(1,1));
- S(2,2)=sqrt(lamda(2,2));
- S % show matrix S
- % Determine U
- AV=A*V
- b1=AV (:,1)/norm(AV(:,1));% normalized vector b1
- b2=AV (:,2)/norm(AV(:,2));% normalized vector b2
- b3=cross(b1,b2); % choose b3 such that U is an orthonormal matrix, e.g. b3=b1xb2
- U=[b1 b2 b3]
- % Check A=USVt?
- disp('Check the result of the decomposition, A=USVt?')
- USVt=U*S*V'
- % you can find out that A'*b3=b3t*A=0; i.e. b3 is normal to each column of A

Implementation with matlab

- A =
- 2 1
- 1 0
- 1 0

- Determine the SVD of matrix A step by step:
- X =
- 6 2
- 2 1

- Eigen vectors V and eigen values lamda of matrix A
- V =
- 0.3310 -0.9436
- -0.9436 -0.3310

- lamda =
- 0.2984 0
- 0 6.7016

- S =
- 0.5463 0
- 0 2.5887
- 0 0
- AV =
- -0.2816 -2.2183
- 0.3310 -0.9436
- 0.3310 -0.9436

- U =
- -0.5155 -0.8569 0
- 0.6059 -0.3645 -0.7071
- 0.6059 -0.3645 -0.7071

- Check the result of the decomposition, A=USVt?
- USVt =
- 2 1
- 1 0
- 1 0

Matlab

Example_svd.m

An Important Property from the SVD

- You may notice that the vector $\mathbf{b}_3$, that we choose to make **U** be an orthonormal matrix, is normal to the columns (vector) of matrix **A**.
- Mathematically speaking, we have

$$\mathbf{A}^T \mathbf{b}_3 = \mathbf{b}_3^T \mathbf{A} = \mathbf{0}$$

- Indeed, we always have $\mathbf{A}^T\mathbf{U}_i=\mathbf{U}_i^T\mathbf{A}=0$. (This implies that the independent vector(s) $\mathbf{U}_i$ span the null space of the matrix $\mathbf{A}^T$.)
- Proof: $\mathbf{A}^T\mathbf{U}_i=\mathbf{U}_i^T\mathbf{A}=0$, $\mathbf{A}=\mathbf{USV}^T$

$$\begin{aligned}\mathbf{A} &= \mathbf{USV}^T \\ &= \begin{bmatrix}\mathbf{U}_d & \mathbf{U}_i\end{bmatrix}\begin{bmatrix}\mathbf{D}\\ \mathbf{0}\end{bmatrix}\mathbf{V}^T \\ &= \mathbf{U}_d\mathbf{D}\mathbf{V}^T\end{aligned}$$

$$\downarrow$$

$$\begin{aligned}\mathbf{U}_i^T\mathbf{A} &= \mathbf{U}_i^T\mathbf{U}_d\mathbf{D}\mathbf{V}^T \\ &= (\mathbf{0})\mathbf{D}\mathbf{V}^T \\ &= \mathbf{0}\end{aligned}$$

$$\mathbf{U}_{m\times m} = \begin{bmatrix}\hat{\mathbf{b}}_1 & \cdots & \hat{\mathbf{b}}_n\end{bmatrix} = \begin{bmatrix}\mathbf{U}_d & \mathbf{U}_i\end{bmatrix}$$

Note: **U** is an orthonormal matrix, thus $\mathbf{U}_d^T\mathbf{U}_i = \mathbf{U}_i^T\mathbf{U}_d = \mathbf{0}$

$$\mathbf{S}_{m\times n} = \begin{bmatrix}\mathbf{D}_{n\times n}\\ \mathbf{0}_{(m-n)\times n}\end{bmatrix}, \mathbf{D}_{n\times n} = \begin{bmatrix}\sqrt{\lambda_1} & & \mathbf{0}\\ & \ddots & \\ \mathbf{0} & & \sqrt{\lambda_n}\end{bmatrix}$$

$$\mathbf{V}_{n\times n}^T = \begin{bmatrix}\mathbf{x}_1 & \cdots & \mathbf{x}_n\end{bmatrix}^T$$

Related Mathematical Background
Null Space

- Definition:
  - Let $A(\mathbf{x})=\mathbf{A}_{m\times n}\mathbf{x}_{n\times 1}$ be a linear transformation from a vector space to another vector space. The null space of matrix $\mathbf{A}_{m\times n}$ is the subspace that consists of all solutions to the equation **Ax** =0. (矩陣**A**之null-space就是所有滿足Ax=0之解(x)所構成的空間, Ex: v=J(dq/dt)=0)
- Thus, for a linear system **Ax** =0, vector **x** lies in the null space of matrix **A**. (若|**A**|≠0, 則矩陣**A**之null-space爲一個點x=0)
- The null space of **A** has dimension *n-m*, where *m* is the rank of **A**. (例如: 聯立方程組中有n個變數,但只有m條線性獨立方程式, 故有n-m個變數可任意指定; 換言之, Ax =0將有$\propto^{n-m}$個解, 且稱這些解所構成之空間的維度爲n-m)

$$\boxed{\Phi_q \dot{\mathbf{q}} = 0 \Leftrightarrow \mathbf{A}\mathbf{x} = 0}$$

- This equation indicates that the allowable velocity vector d**q**/dt at a specific position belongs to the null-space of the Jacobian matrix Φ_q.
- Question: How to determine the null-space of Φ_q? Or how to determine the subspace of the allowable velocities (dq/dt)? 例如: 如何調節robot之三個關節的角速度使其指尖速度等於零.

Physical Interpretation

Φ = x²+y²+z²-L² = 0 ⇔ A particle is constrained to move on the surface of a sphere

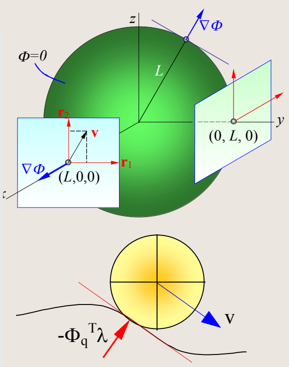
$$d\Phi = \frac{\partial\Phi}{\partial x}dx + \frac{\partial\Phi}{\partial x}dx + \frac{\partial\Phi}{\partial x}dx = \begin{bmatrix} \frac{\partial\Phi}{\partial x} & \frac{\partial\Phi}{\partial y} & \frac{\partial\Phi}{\partial z} \end{bmatrix} \begin{bmatrix} dx \\ dy \\ dz \end{bmatrix} = \Phi_q d\mathbf{q} = 0$$

or

$$d\Phi = \left(i\frac{\partial\Phi}{\partial x} + j\frac{\partial\Phi}{\partial y} + k\frac{\partial\Phi}{\partial z} \right) \bullet (idx + jdy + kdz) = \nabla\Phi \bullet d\mathbf{q} = (\nabla\Phi)^T d\mathbf{q} = 0$$

1. $\nabla\Phi$ is the normal of constraint surface Φ=c, because d**q** is tangential to the surface.
2. $\Phi_q = (\nabla\Phi)^T \Leftrightarrow \Phi_q^T = \nabla\Phi$

- Since $\Phi_q^T = \nabla\Phi$ is the normal of the constraint surface Φ=c, the constraint force exerts on the particle can be written as $-\Phi_q^T\lambda$.
- Since Φ_q**dq**=0, allowable motion **dq** must be in the null-space of the Φ_q.
- Since there is only one constraint (m=1), the dimension of the null-space is 3-1=2; say, the null-space here is the tangent plane of the surface Φ=c
- If there are more than one one constraints, the null space is the intersection (交集) of all constraint surfaces(即同時在多個拘束面上)



Example for Null Space

Matlab Example2_svd.m

- Given: $\Phi = x^2+y^2+z^2-L^2 = 0$ ➔
- Determine: the null space of Φ_q at position **q**=[L, 0, 0]^T

$$\mathbf{\Phi_q} = \begin{bmatrix} 2x & 2y & 2z \end{bmatrix}$$

$$\mathbf{\Phi_q \dot{q}} = \begin{bmatrix} 2x & 2y & 2z \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = 0$$

$$2x\dot{x} + 2y\dot{y} + 2z\dot{z} = 0$$

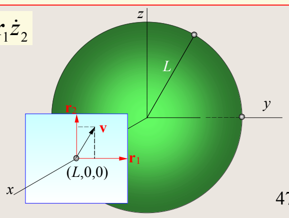
$$\dot{x} = -\frac{y\dot{y} + z\dot{z}}{x}$$

Since the rank of Φ_q is 1, there are (3-1)=2 basis vectors spanning the null space of Φ_q. Choosing (dy/dt, dz/dt) = (1, 0) and (0,1), respectively, yields the following two vectors:

$$\mathbf{r}_1 = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -\frac{y}{x} \\ 1 \\ 0 \end{bmatrix} \qquad \mathbf{r}_2 = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -\frac{z}{x} \\ 0 \\ 1 \end{bmatrix}$$

At **q**=[L, 0, 0]^T, we have $\mathbf{r}_1$=[0, 1, 0]^T , $\mathbf{r}_2$=[0, 0, 1]^T. The null space in this case is a plane(與拘束面相切). All the allowable velocities should be in this plane, and can be expressed as

$$\mathbf{\dot{q}} = \mathbf{r}_1\dot{z}_1 + \mathbf{r}_1\dot{z}_2$$



Example for Null Space

- determine the null space of Φ_q=[1 0 1; 5 4 9; 2 4 6]
- 相當於三條拘束式: Φ_1=x+z=0; Φ_2=5x+4y+9z=0; Φ_3=2x+4y+6z=0

$$\mathbf{\Phi_q \dot{q}} = \begin{bmatrix} 1 & 0 & 1 \\ 5 & 4 & 9 \\ 2 & 4 & 6 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = 0$$

-5r1+r2
c3-c1-c2

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 4 & 4 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = 0$$

$$\begin{aligned} \dot{x} + \dot{z} &= 0 \\ 4\dot{y} + 4\dot{z} &= 0 \end{aligned}$$

$$\therefore \mathbf{\dot{q}} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} -\dot{z} \\ -4\dot{z} \\ \dot{z} \end{bmatrix} = \dot{z} \begin{bmatrix} -1 \\ -1 \\ 1 \end{bmatrix} = \dot{z}\mathbf{r}$$

Since n=3 and the rank of Φ_q is k=2, we can find (n-k)=1 independent vector that belong to the null space of Φ_q.

Note: **r**=[-1 -1 1] is called the basis vector of the null space, it is a line in this case (該線爲面x+z=0和5x+4y+9z=0之交線). Every point in the line corresponds to a solution of Φ_q(dq/dt)=0.

Matlab Example1_svd.m

Numerical Calculating for the Null Space Matrix R

- Let $\mathbf{r}_i$ be a set of $k=n-m$ linearly independent vectors that constitute a basis for the null space of Φ_q; **i.e.,** $(\boldsymbol{\Phi}_q \mathbf{r}_i)=0$. Any velocity vector dq/dt can be expressed as a linear combination of this basis as follows:

$$\begin{aligned}\dot{\mathbf{q}}_{n\times 1} &= \mathbf{r}_1\dot{z}_1 + \mathbf{r}_1\dot{z}_2 + \cdots + \mathbf{r}_k\dot{z}_k \\ &= \begin{bmatrix}\mathbf{r}_1 & \cdots & \mathbf{r}_k\end{bmatrix}_{n\times k}\begin{bmatrix}\dot{z}_1 \\ \vdots \\ \dot{z}_k\end{bmatrix}_{k\times 1} \\ &= \mathbf{R}_{n\times k}\dot{\mathbf{z}}_{k\times 1}\end{aligned}$$

$$\begin{aligned}\mathbf{U}_i^T\mathbf{A} &= \mathbf{U}_i^T\mathbf{U}_d\mathbf{D}\mathbf{V}^T \\ &= (\mathbf{0})\mathbf{D}\mathbf{V}^T \\ &= \mathbf{0}\end{aligned}$$

- Matrix R plays an important role in some of the most efficient formulation for dynamic analysis.
- From the previous discussion, we know that

$$\boxed{\boldsymbol{\Phi}_q\mathbf{r}_i = \mathbf{0} \quad (i=1,\ldots,k)} \longrightarrow \boxed{\boldsymbol{\Phi}_q\begin{bmatrix}\mathbf{r}_1 & \cdots & \mathbf{r}_k\end{bmatrix} = \boldsymbol{\Phi}_q\mathbf{R} = \mathbf{0}}$$

- Matrix R can be determined by calculating the SVD of Φ_q^T
- Let Φ_q^T = A= **USV**^{T} $\Rightarrow$ $\Phi_q\mathbf{U}_i = A^T\mathbf{U}_i = 0$

$\therefore$ **R** = the $\mathbf{U}_i$ in SVD of Φ_q^T

2007/4/4 計算動力學(C.F. Chang) 49