Chapter 3

Type, Number, and Dimensional Synthesis

The science of mechanism kinematics is roughly divided in two divergent topics: analysis and synthesis. Analysis typically involves a defined mechanism and predicting how either the coupler or the follower will react to specific motions of the driver. For example, in Figure 3.1: determine the position of a reference point on the coupler in the fixed frame Σ given input angle ψ and all the link lengths (distances between revolute centers); determine the pose (position and orientation) of reference frame E that moves with the coupler in terms of fixed frame Σ for a given input ψ; determine the change in follower angle ϕ for a change in input angle ψ; or the kinematic equivalent, determine the motion of follower frame F expressed in Σ given an input ψ.

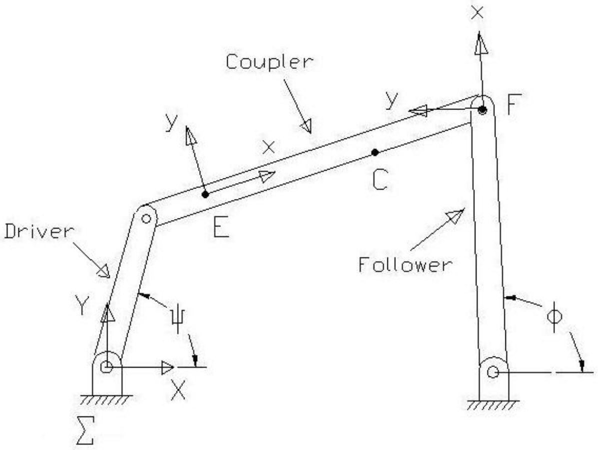


Figure 3.1: 4R four-bar mechanism.

We will look at analysis later on in the context of robot kinematics. The problems are innately geometric, hence we shall adopt a geometric approach.

A fundamentally different problem is that of kinematic synthesis. But, as we shall see, with some geometric insight, we can use the same approach as for analysis. By *kinematic synthesis* we mean the design or creation of a mechanism to attain specific motion characteristics. In this sense synthesis is the inverse problem of analysis. Synthesis is the very essence of design because it represents the creation of new hardware to meet particular requirements of motion: displacement; velocity; acceleration; individually or in combination. Some typical examples of kinematic synthesis include:

1. Guiding a point along a specific curve. Examples include Watt's "straight" line linkage, post-hole diggers, etc. See Figure 3.2.

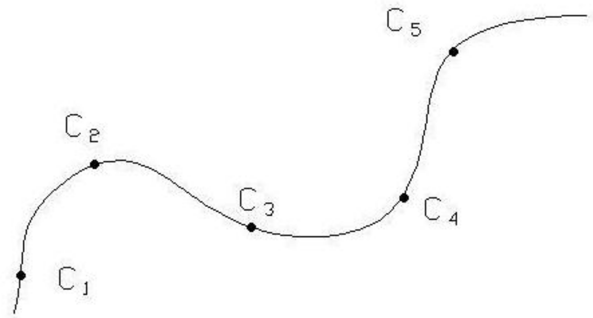


Figure 3.2: Point C is guided along the curve to sequential positions C_1 to C_5.

2. Correlating driver and follower angles in a functional relationship: the resulting mechanism is a *function generator*. In essence the mechanism generates the function $\phi = f(\psi)$, or vice versa. See Figure 3.3. The classic example of a function generator is a four-bar steering linkage. The function is the *steering condition*, and is defined by:

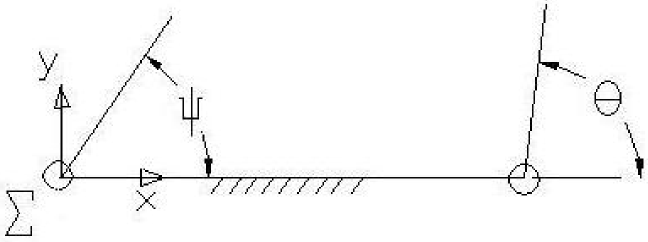


Figure 3.3: Function generator.

$$S(\Delta\psi\Delta\phi) \equiv \sin(\Delta\phi - \Delta\psi) - \rho\sin(\Delta\psi)\sin(\Delta\phi) \quad = \quad 0. \qquad (3.1)$$

In the steering condition $\Delta\psi$ and $\Delta\phi$ are the change in input and output angles, respectively from the input and output *dial zeroes*, α and β, respectively (i.e., $\psi = 0$ and $\phi = 0$ are not necessarily in the direction of the x-axis, they are offset). The quantity ρ is the length ratio b/a, with a the distance between axles and b the distance between wheel-carrier revolutes, see Figure 3.4.

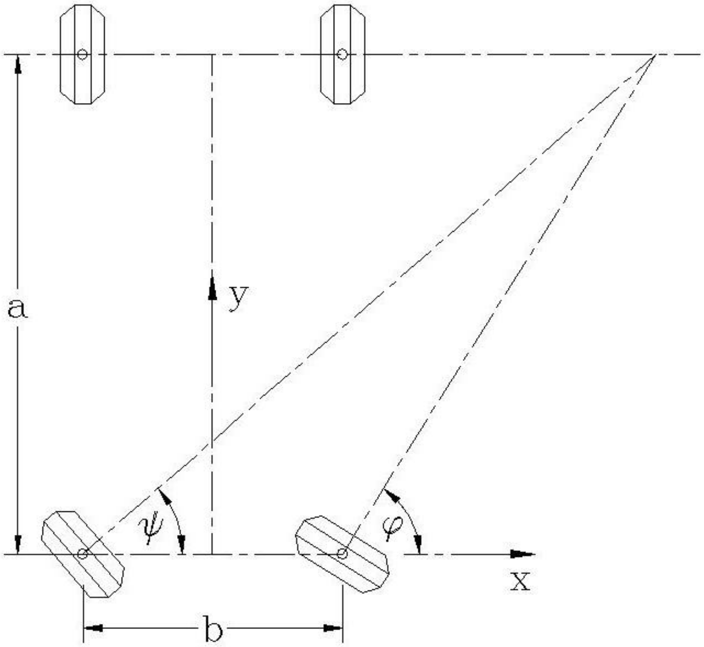


Figure 3.4: The steering condition.

3. Guiding a rigid body through a finite set of positions and orientations, as in Figure 3.5. A good example is a landing gear mechanism which must retract and extend the wheels, having down and up locked poses (position and orientation) with specific intermediate poses for collision avoidance. See Figure 3.6. Synthesis for rigid body guidance is also known as the *Burmester Problem*, after Ludwig Burmester (1840-1927), Professor of Descriptive Geometry and Kinematics. His book *Lehrbuch der Kinematik* (1888), is a classic systematic and comprehensive treatment of theoretical kinematics.

4. Trajectory Generation: Position, velocity, and/or acceleration must be correlated along a curve. Examples include film transport mechanisms in motion picture projectors. Another example is the 4-bar linkage designed to emulate a rack-and-pinion drive, shown in Figure 3.7. If the input

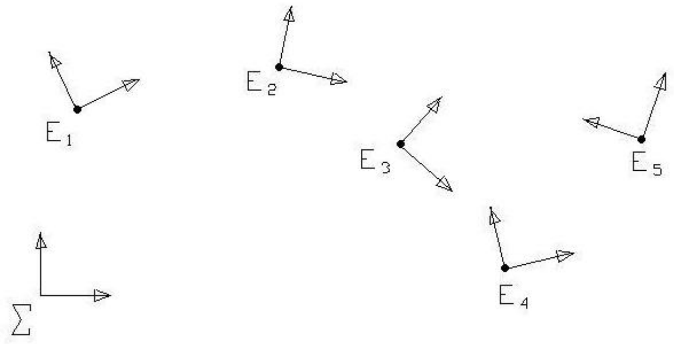


Figure 3.5: Rigid body guidance (motion generation).

Figure 3.6: Landing gear.
www.aerospace-technology.com/contractor_images/finnair/LandingGear_5s.jpg

pinion angular velocity is constant then the linear velocity of the rack will also be constant along a straight line. The displacement of the rack is *timed* to that of the pinion. The linkage shown in the Figure 3.8 produces approximately straight line motion along a section of its coupler curve. Moreover the position of the coupler point R is approximately timed to input angle ψ along the "straight" section, hence $\dot{\psi}$ is linearly proportional to $\dot{R}$ on that section.

Note that the coupler curve is endowed with central line symmetry (two halves are reflected in a line). The coupler point R generating this symmetric coupler curve is on a circle centered at C and passing though A. The coupler point is on the same circle $\Rightarrow |AC| = |CR| = |CD|$.

The design of a device to achieve a desired motion involves three distinct synthesis problems: type, number, and dimensional synthesis. These three prob-

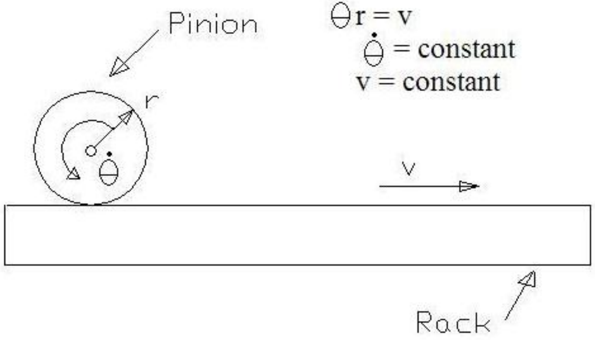


Figure 3.7: Rack-and-pinion drive.

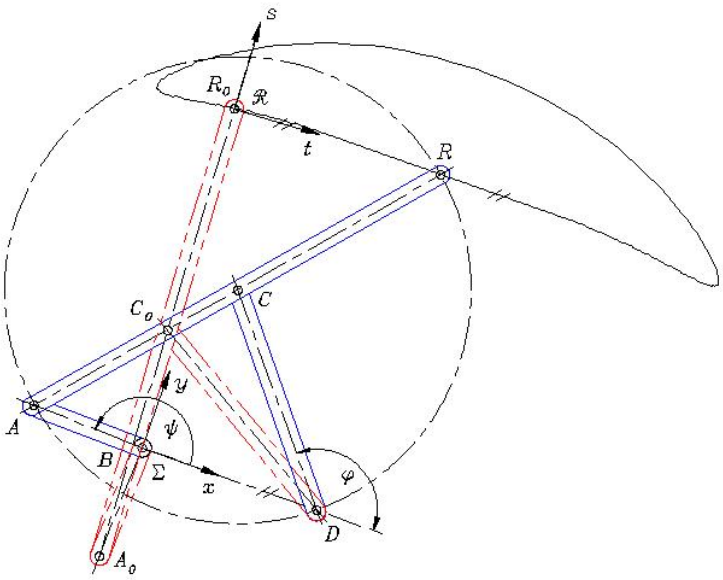


Figure 3.8: The line generating linkage where $\dot{\psi}$ is proportional to $\dot{R}$ along the line. The coupler curve is central line symmetric. This implies the coupler and follower have the same length: distance between centers A and C is equal to the distance between C and D. This is also equal to the distance between C and coupler point R, while A, C, and R are collinear.

lems will be individually discussed in the following three sections.

3.1 Type Synthesis

Machines and mechanisms involve assemblies made from six basic types of mechanical components. These components were identified by the German kinematician Franz Reuleaux in his classic book from 1875 *Theroetische Kinematik*, translated into English in 1876 by Alexander Blackie William Kennedy (of Aronhold-Kennedy Theorem fame) as *Kinematics of Machines.* The six categories are:

1. Eye-bar type link (the lower pairs).

2. Wheels, including gears.

3. Cams in their many forms.

4. The screw (which transmits force and motion).

5. Intermittent-motion devices (rachets).

6. Tension-compression parts with one-way rigidity such as belts, chains, and hydraulic or pneumatic lines.

The section of the type of mechanism needed to perform a specified motion depends to a great extend on design factors such as manufacturing processes, materials, safety, reliability, space, and economics, which are arguably outside the field of kinematics.

Even if the decision is made to use a planar 4-bar mechanism, the *type* issue is not laid to rest. There remains the question: "What type of 4-bar?". Should it be an RRRR? An RRRP? An RPPR? There can be 16 possible planar chains comprising R- and P-Pairs. Any one of the four links joined by the four joints can be the fixed link, each resulting in a different motion. See Figure 3.9.

We will look at an interesting result for type synthesis, otherwise we will leave it alone.

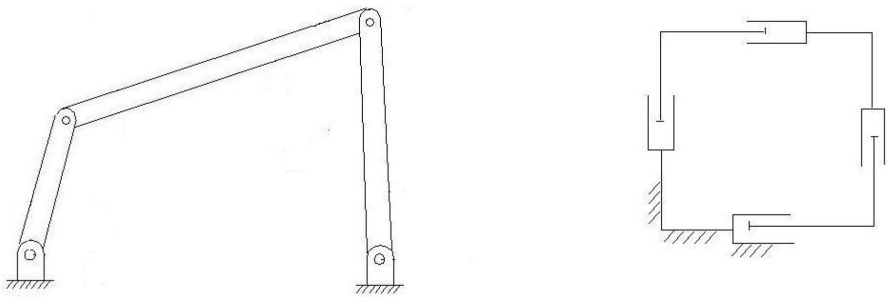
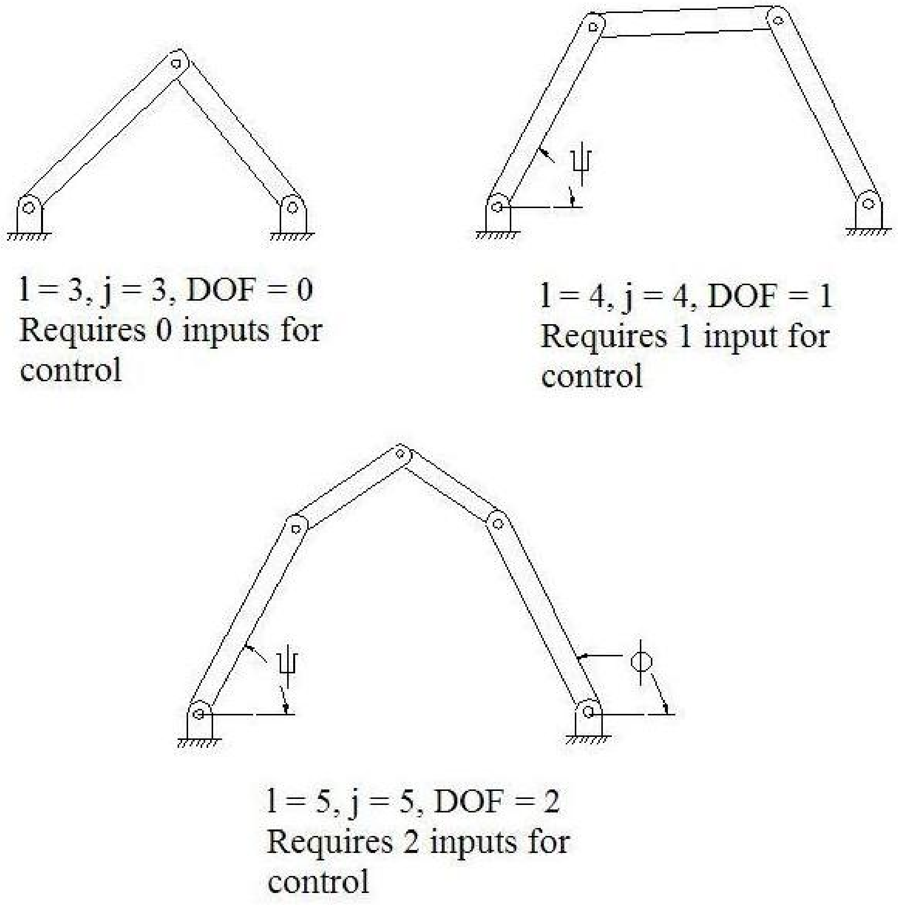


Figure 3.9: 16 distinct ways to make a planar 4-bar using only R- and P-pairs taken 4 at a time. The extreme cases RRRR and PPPP are shown.

3.2 Number Synthesis

Number synthesis is the second step in the process of mechanism design. It deals with determining the number of DOF and the number of links and joints required. See Figure 3.10. If each DOF of the linkage is to be controlled in a closed kinematic chain, then an equal number of active inputs is required. If the linkage is a closed chain with 2 DOF then actuators are required at two joints. If more actuators are used the device is *redundantly actuated*, if less are used, the device is *under actuated* with uncontrolled DOF.



l = 3, j = 3, DOF = 0
Requires 0 inputs for control
ψ
l = 4, j = 4, DOF = 1
Requires 1 input for control
ψ
ϕ
l = 5, j = 5, DOF = 2
Requires 2 inputs for control

Figure 3.10: Linkages with 0,1,and 2 DOF, requiring 0,1,and 2 inputs to control each DOF.

Open kinematic chains require that each joint be actuated. A 7R planar open chain has 3 DOF, but requires 7 actuators to control the 3 DOF, although it is still said to be redundantly actuated. See Figure 3.11. Note that in the

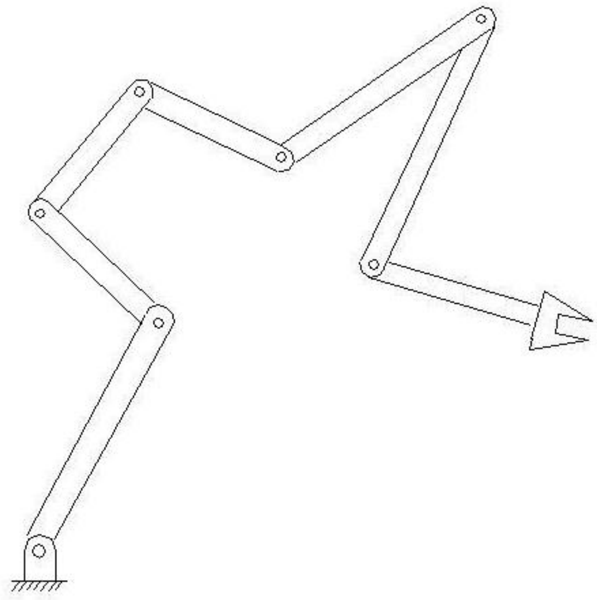


Figure 3.11: Planar 7R with 3 DOF, even though CGK says 7 DOF.

7R open planar chain example, the CGK formula predicts 7 DOF:

$$\begin{aligned} 3(8-1) - 7(2) = 21 - 14 &= 7DOF, \\ 6(8-1) - 7(5) = 42 - 35 &= 7DOF. \end{aligned}$$

The 7 DOF in E_2 and E_3 are indeed real, but linearly coupled. There is a maximum of 3 DOF in the plane, and a maximum of 6 DOF in Euclidean space.

3.3 Dimensional Synthesis

The third step in mechanism design is dimensional synthesis. We will consider three distinct problem classes in dimensional synthesis mentioned earlier, namely: function generation; rigid body guidance; and trajectory generation.

3.3.1 Function Generation

Dimensional synthesis boils down to solving a system of synthesis equations that express the desired motion in terms of the required unknown dimensions. These unknowns include, among other things, the link lengths, relative locations of revolute centers, and in particular for function generators, a *zero* or *home* configuration.

There is an old joke that goes: The way to tell the difference between a mathematician and an engineer is to give them a problem. The engineer will immediately proceed to seek a solution; the mathematician will first try to prove that a solution exists. Engineers laugh at mathematicians and vice versa when the joke is told. The history of mathematics contains many stories of wasted effort attempting to solve problems that were ultimately proved to be unsolvable. Problems of *existence* should not be ignored.

The existence of a solution must be examined for linear systems before we proceed to attempt to solve. We must ask two questions:

1. Is the system consistent; does at least one solution exist?
2. If one solution exists, is it unique?

We can always represent a set of m linear equations in n unknowns as:

$$\vec{X}_{m\times 1} = \tilde{A}_{m\times n}\vec{x}_{n\times 1} \tag{3.2}$$

$\vec{X}$ is an $m \times 1$ column vector, $\tilde{A}$ is a matrix with m rows and n columns, and $\vec{x}$ is an $n \times 1$ column vector. In general, we can make the following claims:

1. If there are fewer equations than unknowns ($m < n$), then if one solution exists, and infinite number of solutions exist.
2. If the number of equations equals the number of unknowns ($m = n$), then if a solution exists it is unique.
3. If there are more equations than unknowns ($m > n$), then, in general, no solutions exists.

3.3.2 Basic Linear Algebra Concepts

Vector, Array, and Matrix Operations

While Chinese mathematicians had developed algorithms for solving linear systems by 250 BC, linear algebra as we know it was first formalized by German mathematician and Sanskrit scholar Hermann Grassmann (1808-77). The new algebra of vectors was presented in his book *Die Ausdehnungslehre* (1844).

Linear Dependence

Consider three equations linear in three unknowns of the form:

$$Ax = b$$

One way to test if the three equations are linearly independent (no line common to all three planes) is to look at the coefficient matrix A. The system is linearly dependent if at least two rows (or two columns) are scalar multiples of each other.

$$\begin{bmatrix} 5 & -1 & 2 \\ -2 & 6 & 9 \\ -10 & 2 & -4 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 7 \\ 0 \\ 18 \end{bmatrix},$$

Solutions to Linear Systems

Finding the solution(s) of m equations linear in n unknowns boils down to the following three classes:

1. Underdetermined: $(m < n)$ If a finite solution exists, it is not unique. In fact, if one exists, then an infinite number exists.

2. Overdetermined: $(m > n)$ A finite solution(s) may exist, but not in general.

3. Determined System: $(m = n)$ A unique solution may exist, although it may not be finite. Additionally there may be infinitely many, or no finite solutions.

In 3D Euclidean space, every system of linear equations has either: no solution (the system is inconsistent); exactly one solution (the system is consistent); or infinitely many solutions (the system is consistent). For now we shall concentrate on determined systems: three equations linear in three unknowns. Geometrically, one linear equation in three unknowns (x, y, z) represents a *plane* in the *space* of the unknowns.

Taking the three equations to represent three planes in Euclidean 3D space, solutions to the system are bounded by Euclid's *parallel axiom*: Given a line L and a point P, not on L, there is one and only one line through P that is parallel to L (and two parallel lines do not intersect). Extending this axiom to planes, three planes either:

1. have no point in common;

2. have exactly one point in common;

3. have infinitely many points in common (the planes have a line in common, or are all incident)

If we extend 3-D Euclidean space to include all points at infinity, things change. Now every parallel line intersects in a point on a line at infinity, and every parallel plane intersects in a line on the plane at infinity. In this sense, there are five possibilities for a system of three equations linear in three unknowns:

1. unique finite solution;

2. infinite finite solution;

3. double infinity of finite solutions;

4. unique solution at infinity;

5. infinite solutions at infinity (occurs in two ways).

1: Unique Finite Solution: Consider the linear system:

$$\begin{aligned} 5x - y + 2z &= 7, \\ -2x + 6y + 9z &= 0, \\ -7x + 5y - 3z &= -7. \end{aligned}$$

This system can be represented ($Ax = b$) as:

$$\begin{bmatrix} 5 & -1 & 2 \\ -2 & 6 & 9 \\ -7 & 5 & -3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 7 \\ 0 \\ -7 \end{bmatrix}.$$

Solution (using Matlab "\" operator):

$$x = A \setminus b$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = (1/13) \begin{bmatrix} 21 \\ 10 \\ -2 \end{bmatrix},$$

There is only one possible solution, the three planes intersect at a common point. See Figure 3.12.

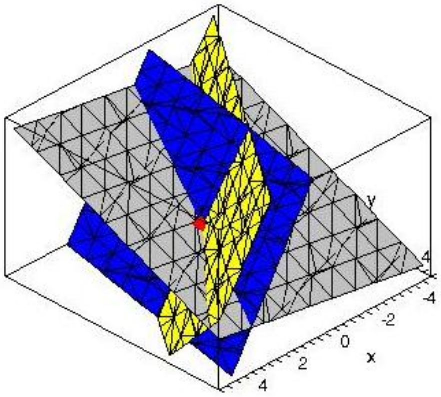


Figure 3.12: One real solution.

2: Infinite Solutions Along a Finite Line: Consider the linear system:

$$\begin{aligned} 5x - y + 2z &= 7, \\ -2x + 6y + 9z &= 0, \\ 15x - 3y + 6z &= 21. \end{aligned}$$

This system can be represented ($Ax = b$) as:

$$\begin{bmatrix} 5 & -1 & 2 \\ -2 & 6 & 9 \\ 15 & -3 & 6 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 7 \\ 0 \\ 21 \end{bmatrix},$$

Solution:

$$x = A \setminus b$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = (1/4) \begin{bmatrix} 6 - 3t \\ 2 - 7t \\ 4t \end{bmatrix},$$

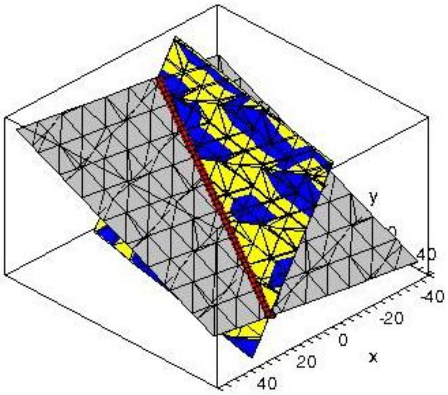


Figure 3.13: Infinitely many real solutions. One parameter family of points.

Two of the three planes are superimposed as shown in Figure 3.13. There will be an infinite number of solutions along the line of intersection between the superimposed planes and the third, intersecting plane.

3: Infinite Solutions on a Finite Plane: Consider the linear system:

$$\begin{aligned} 5x - y + 2z &= 7, \\ 10x - 2y + 4z &= 14, \\ -15x + 3y - 6z &= -21. \end{aligned}$$

This system can be represented ($Ax = b$) as:

$$\begin{bmatrix} 5 & -1 & 2 \\ 10 & -2 & 4 \\ -15 & 3 & -6 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 7 \\ 14 \\ -21 \end{bmatrix},$$

Solution:

$$x = A \setminus b \;\; \rightarrow \;\; 5x - y + 2z = 7$$

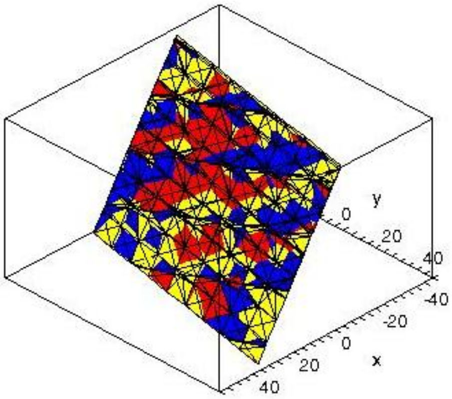


Figure 3.14: Infinitely to the power 2 solutions.

All three planes are superimposed as in Figure 3.14. There will be an infinite number of solutions: a two parameter family of lines covering the entire plane.

4: Infinite Solutions at Infinity: Consider the linear system:

$$\begin{aligned} 5x - y + 2z &= 7, \\ -10x + 2y + -4z &= -5, \\ 15x - 3y + 6z &= -5. \end{aligned}$$

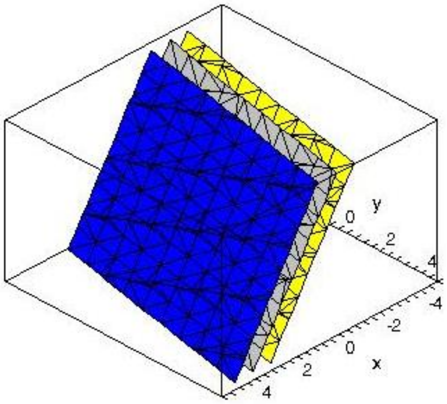


Figure 3.15: One solution where the line at infinity of the blue and yellow planes intersects the grey plane.

This system can be represented $(Ax = b)$ as:

$$\begin{bmatrix} 5 & -1 & 2 \\ -10 & 6 & -4 \\ 15 & -3 & 6 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 7 \\ -5 \\ -5 \end{bmatrix},$$

Solution:

$$x = A \setminus b$$

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \infty,$$

Three parallel non-coincident planes as in Figure 3.15. The three planes all intersect the plane at infinity in the same real, but not finite line: a line on the plane at infinity.

5: No Finite (One Infinite) Solution: Consider the linear system:

$$\begin{aligned} 5x - y + 2z &= 7, \\ -2x + 6y + 9z &= 0, \\ -10x + 2y - 4z &= 18. \end{aligned}$$

This system can be represented $(Ax = b)$ as:

$$\begin{bmatrix} 5 & -1 & 2 \\ -2 & 6 & 9 \\ -10 & 2 & -4 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 7 \\ 0 \\ 18 \end{bmatrix},$$

Solution:

$$x = A \setminus b$$

$$\begin{aligned} 5x - y + 2z &= 7, \\ 5x - y + 2z &= -9. \end{aligned}$$

There are two parallel (linearly dependent, but not coincident) planes, intersecting a third plane as in Figure 3.16. There will never be a finite point/line/plane where the system of equations will intersect.

6: No Finite Solution: Consider the linear system:

$$\begin{aligned} 5x - y + 2z &= -12, \\ -2x + 6y + 9z &= 0, \\ 8y + 14z &= 8. \end{aligned}$$

This system can be represented $(Ax = b)$ as:

$$\begin{bmatrix} 5 & -1 & 2 \\ -2 & 6 & 9 \\ 0 & 8 & 14 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} -12 \\ 0 \\ 8 \end{bmatrix},$$

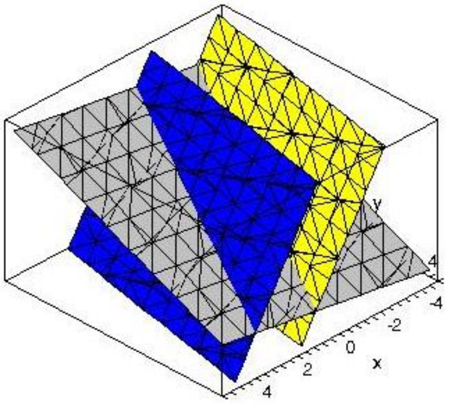


Figure 3.16: Lines of intersection of each pair are parallel. One unique solution at infinity. One solution where the line at infinity of the blue and yellow planes intersects the grey plane.

> Solution: Three non-intersecting planes as shown in Figure 3.17. The line of intersection of each pair of planes is parallel to the third. The parallel lines intersect in a real point at infinity.

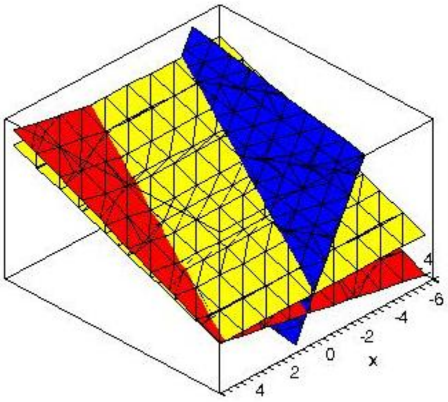


Figure 3.17: No Finite Solution: three non-intersecting planes.

3.3.3 Dimensional Synthesis Continued

The synthesis equations for a 4-bar function generator that are currently used were developed by Ferdinand Freudenstein in his Ph.D thesis in 1954. Consider the mechanism in the Figure 3.18. The i^{th} configuration is governed by:

$$k_1 + k_2 \cos(\phi_i) - k_3 \cos(\psi_i) \quad = \quad \cos(\psi_i - \phi_i). \tag{3.3}$$

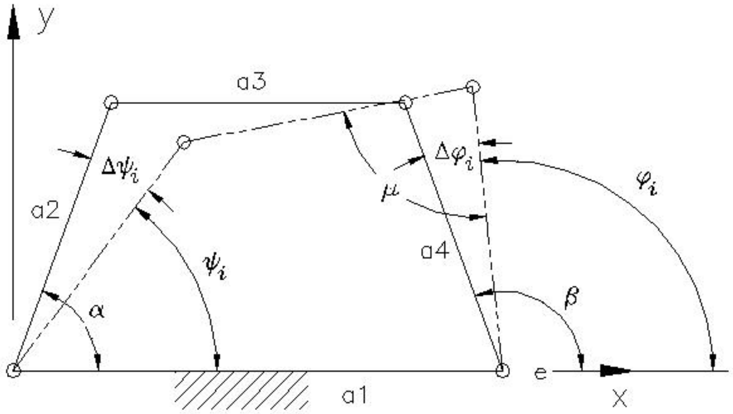


Figure 3.18: Elements of the Freudenstein parameters for a planar 4R linkage.

Where the k's are the *Freudenstein Parameters* which are the following link ratios:

$$\begin{aligned}
k_1 &\equiv \frac{(a_1^2+a_2^2+a_4^2-a_3^2)}{2a_2a_4},\\
k_2 &\equiv \frac{a_1}{a_2},\\
k_3 &\equiv \frac{a_1}{a_4},\\
\Rightarrow a_1 &= 1,\\
a_2 &= \frac{1}{k_2},\\
a_4 &= \frac{1}{k_3},\\
a_3 &= (a_1^2+a_2^2+a_4^2-2a_2a_4k_1)^{1/2}. \qquad (3.4)
\end{aligned}$$

These distinct sets of input and output angles (ψ, ϕ) define a fully determined set of synthesis equations, implying that if there is a solution of k_1, k_2, k_3 that satisfies the three equations, it will be unique. Thus, we can determine the link length ratios (k_1, k_2, k_3) that will exactly satisfy the desired function $\phi = f(\psi)$. Of course for a function generator we only require ratios of the links because the scale of the mechanism is irrelevant.

What happens if you need a function generator to be exact over a continuous range of motion of the mechanism? Sadly, you can't have it. A solution, in general, does not exist! Now do you hear the gentle chuckles of the mathematicians? To be exact in greater than three configurations results in an over-determined system (more equations than unknowns), which in general has no solution. We could try to make an improvement by formulating the problem so that there were more unknowns. Recall the steering condition, reproduced here:

$$S(\Delta\psi\Delta\phi) \equiv \sin(\Delta\phi - \Delta\psi) - \rho\sin(\Delta\psi)\sin(\Delta\phi) \quad = \quad 0. \tag{3.5}$$

Define the *dial zeroes* to be α and β such that:

$$\begin{aligned} \psi_i &= \alpha + \Delta\psi_i, \\ \phi_i &= \beta + \Delta\phi_i, \end{aligned} \tag{3.6}$$

The synthesis equation now becomes:

$$k_1 + k_2\cos(\beta + \Delta\phi_i) - k_3\cos(\alpha + \Delta\psi_i) = \cos\left((\alpha + \Delta\psi_i) - (\beta + \Delta\phi_i)\right). \tag{3.7}$$

Now five sets of $(\Delta\psi_i, \Delta\phi_i)$ will produce five equations in five unknowns. This means that the function will be generated exactly for five inputs only. There is no guarantee about the values in between the *precision* input-output values of the function. If this was how steering linkages were designed we would wear out our tires very fast, and have difficulty cornering.

Note, the design constant ρ must be selected, where $\rho = b/a$, with $b =$ track (distance between wheel carrier revolutes), and $\alpha =$ wheelbase (distance between axle center-lines). A typical value for a car is $\rho = 0.5$. We will use this value in the following example shown in Figure 3.19. To resolve the problem, we resort to *approximate synthesis*, instead of *exact synthesis*. This involves over-constraining the synthesis equation. We generate a set of m equations in n unknowns with $m \gg n$. In our case $n = 5$. Two questions immediately arise: What value should be given to m? And how should the input data be distributed? The answer to both questions depends on context, and usually requires some experimentation.

Assuming we have selected m and a distribution for the m input-output (I/O) values, we want a solution that will generate the function with the least error. The *design* and *structural* errors are two performance indicators used to optimize approximate synthesis.

3.4 Design Error

Given an over-constrained system of linear equations (an over-abundance of equations compared to unknowns), in general the system is inconsistent. If dial zeroes α and β have been determined, then we have the three unknown Freudenstein parameters $[k_1, k_2, k_3]^T = \vec{k}$ that must satisfy the $m \gg 3$ equations. Since, in general, no $\vec{k}$ can satisfy all the equations exactly, the design error is defined as:

$$\vec{d} \quad = \quad \vec{b} - \tilde{S}\vec{k}. \tag{3.8}$$

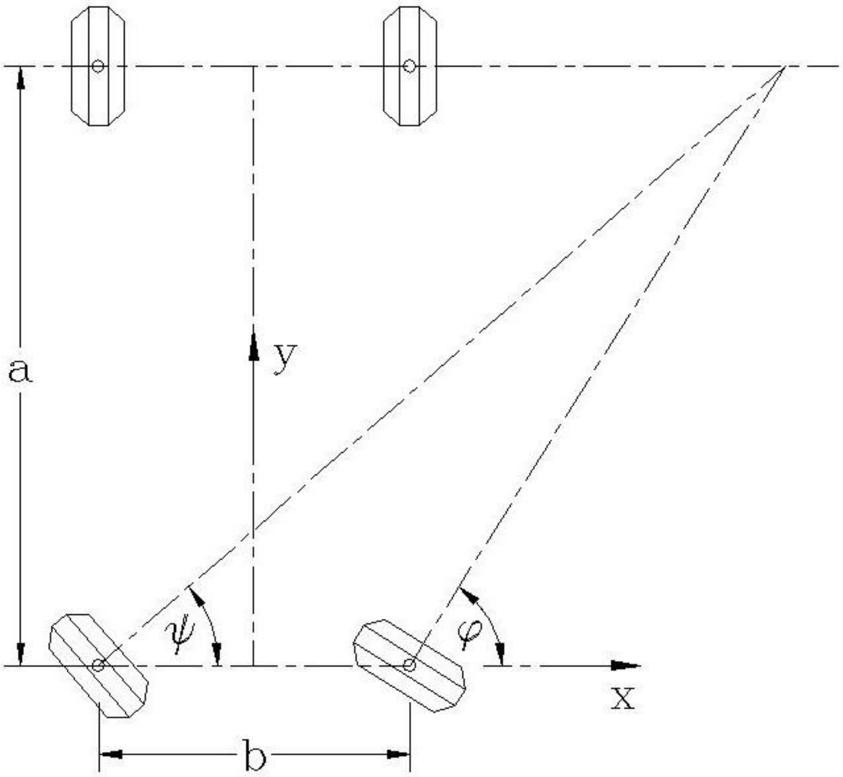


Figure 3.19: The steering condition.

where for the m equations $\tilde{S}$ is an $m \times 3$ matrix (called the synthesis matrix)

$$\tilde{S} = \begin{bmatrix} 1 & \cos(\phi_1) & -\cos(\psi_1) \\ 1 & \cos(\phi_2) & -\cos(\psi_2) \\ . & . & . \\ . & . & . \\ 1 & \cos(\phi_m) & -\cos(\psi_m) \end{bmatrix}$$

$\vec{b}$ is the $m \times 1$ vector

$$\vec{b} = \begin{bmatrix} \cos(\psi_1 - \phi_1) \\ \cos(\psi_2 - \phi_2) \\ . \\ . \\ \cos(\psi_m - \phi_m) \end{bmatrix} \tag{3.9}$$

$\vec{k}$ is the 3×1 vector of unknown Freudenstein parameters, and $\vec{d}$ is the $m \times 1$ design error vector. The Euclidean norm of $\vec{d}$ is a measure of how well the computed $\vec{k}$ satisfies the vector equation

$$\tilde{S}\vec{k} = \vec{b}$$

The objective function is

$$z = (1/2)(\vec{d}^T W \vec{d}). \tag{3.10}$$

which must be minimized over $\vec{k}$. The quantity $(\vec{d}^T W \vec{d})^{1/2}$ is the *weighted Euclidean norm.* This requires a few words of explanation.

The matrix $\tilde{W}$ is a diagonal matrix of positive weighting factors. They are used to make certain data points affect the minimization more, or less, than others depending on their relative importance to the design. We usually start by setting $W = I$ (the identity matrix).

Vector norms serve the same purpose on vector spaces that the absolute value does on the real line. They furnish a measure of distance. More precisely, the vector space R^n together with a norm on R^n define a *metric space.* Since we will be comparing metric and non-metric geometries, we'd better have a definition of *metric space.*

Cartesian Product: Of any two sets S and T, denoted $S \times T$, is the set of all ordered pairs (s, t) such that $s \in S$ and $t \in T$.

Definition of a Metric: Let S be any set. A function d from $S \times S$ into R(the set of real numbers)

$$R = d(S \times S) \equiv d_{s_i s_j}$$

is a metric on S if:

1. $d_{xy} \geq 0, \forall\ (x, y) \in S$;
2. $d_{xy} = d_{yx}, \forall\ (x, y) \in S$;
3. $d_{xy} = 0$ iff $x = y$;
4. $d_{xz} + d_{zy} \geq d_{xy}, \forall\ (x, y, z) \in S$.

Metric Space: A metric space is a non-empty set S, together with a metric d defined on S. For example, the Euclidean distance function for orthogonal x-y-z system:

$$d_{12} = d_{21} = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}.$$

3.5 General Vector Spaces

Definition: Let V be an arbitrary non-empty set of objects on which two operations are defined: addition and multiplication by scalars (real numbers). If the following axioms are satisfied by all objects $\vec{u}, \vec{v}, \vec{w}$ in V and all scalars k and l then V is a *vector space* and the objects in V are vectors:

1. if $\vec{u}, \vec{v} \in V$, then $\vec{u} + \vec{v} \in V$,

2. $\vec{u}+\vec{v}=\vec{v}+\vec{u}$,
3. $\vec{u}+(\vec{v}+\vec{w})=(\vec{u}+\vec{v})+\vec{w}$,
4. $\exists\,\vec{0}\in V$ such that $\vec{0}+\vec{v}=\vec{v}+\vec{0}=\vec{v}$,
5. $\vec{u}+(-\vec{u})=(-\vec{u})+\vec{u}=\vec{0}$,
6. $k\vec{u}\ \in V$, $\forall\, k\ \in R$ and $\forall\,\vec{u}\ \in V$,
7. $k(\vec{u}+\vec{v})=k\vec{u}+k\vec{v}$,
8. $(k+l)\vec{u}=k\vec{u}+l\vec{u}$,
9. $k(k\vec{u})=(kl)\vec{u}$,
10. $1\vec{u}=\vec{u}$.

3.5.1 Vector Norms

A useful class of vector norms are the *p-norms* defined by:

$$||\vec{x}||_p \quad = \quad (|x_1|^p+...+|x_n|^p)^{1/p}. \tag{3.11}$$

Of these the 1,2, and ∞ norms are frequently used:

$$\begin{aligned} ||\vec{x}||_1 &= |x_1|+...+|x_n|; \\ ||\vec{x}||_2 &= (|x_1|^2+...+|x_n|^2)^{1/2} \\ &= (\vec{x}\cdot\vec{x})^{1/2} \\ &= (\vec{x}^{\,T}\vec{x})^{1/2}; \\ ||\vec{x}||_\infty &= \max_{1<i<n}|x_i|. \end{aligned}$$

Example: Consider the vector in a four dimensional vector space:

$$\begin{aligned} x &= \begin{bmatrix} 2 \\ 1 \\ -2 \\ 4 \end{bmatrix}, \\ ||\vec{x}||_1 &= 2+1+2+4 \ = \ 9, \\ ||\vec{x}||_2 &= (4+1+4+16)^{1/2} \ = \ 5, \\ ||\vec{x}||_\infty &= 4. \end{aligned}$$

The 2-norm is the generalized Euclidean distance and is geometrically very intuitive. Back to the objective function for minimizing $\vec{d}$ over $\vec{k}$:

$$z=(1/2)\vec{d}^T\tilde{W}\vec{d}.$$

Typically, one half of the square of the weighted 2-norm is used. The weighted 2-norm (Euclidean norm) of $\vec{d}$ over $\vec{k}$ can be minimized, in a least-squares sense by transforming $\tilde{S}$ using *householder reflections*, or singular value decomposition (SVD). We will discuss these later on.

Briefly, in MATLAB householder orthogonalization is invoked when a rectangular matrix is *divided* by a vector, using the syntax

$$k = s \setminus b.$$

Be warned that this symbolic matrix division is *literally* symbolic: the operation does not exist! The *backslash* operator invokes an algorithm that employs Householder orthogonalization of the matrix. Using SVD is somewhat more involved, but information on how well each element of $\vec{k}$ has been estimated is a direct, and extremely useful output of the decomposition.

Design Error:

$$\vec{d} = \vec{b} - \tilde{S}\vec{k}$$

$$\begin{aligned} \tilde{S} &= \begin{bmatrix} 1 & \cos\phi_1 & -\cos\psi_1 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ 1 & \cos\phi_m & -\cos\psi_m \end{bmatrix}, \\ \vec{k} &= \begin{bmatrix} k_1 \\ k_2 \\ k_3 \end{bmatrix}, \\ \vec{b} &= \begin{bmatrix} \cos(\psi_1 - \phi_1) \\ \cdot \\ \cdot \\ \cos(\psi_m - \phi_m) \end{bmatrix}. \end{aligned}$$

Objective Function:

$$z \;\equiv\; (1/2)(\vec{d}^{\,T}\tilde{W}\vec{d}) \;\Rightarrow\; \text{minimize with respect to } \vec{k}.$$

Normality Condition:

$$\frac{dz}{d\vec{k}} \;=\; 0 \;\Rightarrow\; \begin{bmatrix} (\delta z)/(\delta k_1) \\ (\delta z)/(\delta k_2) \\ (\delta z)/(\delta k_3) \end{bmatrix} \;=\; 0.$$

$$\begin{aligned} \text{Chain rule: } \frac{dz}{d\vec{k}} \quad &\Rightarrow \quad \frac{dz}{dk_i} = \frac{dz}{dd_j}\frac{dd_j}{dk_i} \\ &\Rightarrow \quad \frac{dz}{d\vec{k}} = \left(\frac{d\vec{d}}{d\vec{k}}\right)^T \frac{dz}{d\vec{k}}. \end{aligned}$$

Thus:

$$\begin{aligned} \frac{dz}{d\vec{k}} &= -\tilde{S}^T\tilde{W}\vec{d}, \\ &= -\tilde{S}^T\tilde{S}(\vec{b} - \tilde{s}\vec{k}) \;=\; 0, \\ \Rightarrow\; -\tilde{S}^T\tilde{W}\vec{b} + \tilde{S}^T\tilde{W}\tilde{S}\vec{k} &= 0 \;\Rightarrow\; \text{Normal Equations}, \\ \Rightarrow\; (\tilde{S}^T\tilde{W}\tilde{S})_{(3\times 3)}\vec{k}_{(3\times 1)} &= (\tilde{S}^T\tilde{W})_{(3\times m)}\vec{b}_{(m\times 1)}. \end{aligned}$$

Hence,

$$\vec{k}_{opt} = (\tilde{S}^T\tilde{W}\tilde{S})^{-1}\tilde{S}^T\tilde{W}\vec{b}.$$

Because $\tilde{W} > 0 \;\Rightarrow\; \tilde{S}^T\tilde{W}\tilde{S} > 0 \;\Rightarrow\; K_{opt}$ minimizes z. The term $(\tilde{S}^T\tilde{W}\tilde{S})^{-1}\tilde{S}^T\tilde{W}$ in the above equation is called the *Moore-Penrose Generalized Inverse* (M-PGI) of $\tilde{S}$.

Calculating the Moore-Penrose Generalized Inverse *conceptually* leads to $\vec{k}_{opt}$, however, this is generally not the case numberically, for $m \gg n$, $(\tilde{S}^T \tilde{W} \tilde{S})^{-1} \tilde{S}^T \tilde{W}$ is typically ill-conditioned $\left(\frac{\sigma_{max}}{\sigma_{min}} \approx \infty \right)$.

We *must* resort to a numerical approximation that is insensitive to the conditioning of the M-PGI of $\tilde{S}$. Geometrically, we must find $\vec{k}$ whose image under $\tilde{S}$ is as close possible to $\vec{b}$. This implies the use of Householder Reflections (more later). See Figure 3.20.

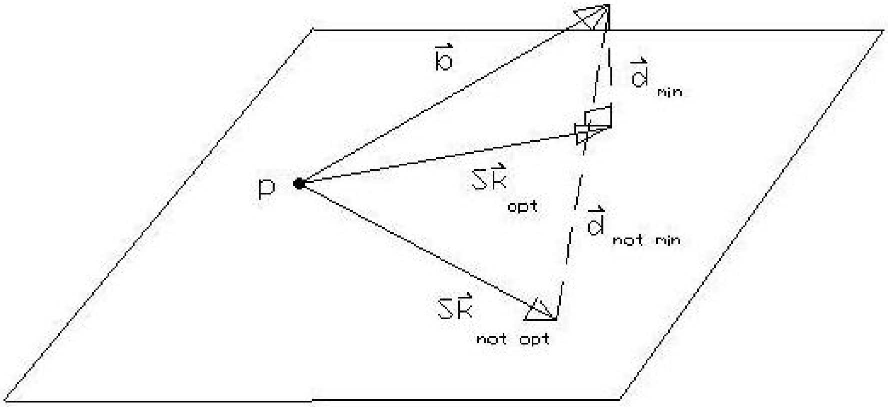


Figure 3.20: $\vec{k}$ rotates $\tilde{S}\vec{k}$ about $\vec{0} = \vec{0}$. $\vec{k}_{opt}$ yields smallest distance (d_{min} between $\tilde{S}\vec{k}$ and $\vec{b}$.

3.6 Structural Error

The structural error is defined to be the difference between the output as specified by the prescribed and generated values for a given input. Let the structural error vector $\vec{s}$ be defined as

$$\vec{s} \equiv [\phi_{pres,i} - \phi_{gen,i}], \ i \ \in \ 1, 2, .., m.$$

To minimize the Euclidean, or 2-Norm, of $\vec{s}$ requires the iterative Gauss-Newton procedure. The scalar objective function to be minimized over $\vec{k}$ is

$$\zeta = (1/2)(\vec{s}^T \tilde{W} \vec{s}).$$

Here, again $\tilde{W}$ is a diagonal matrix of weighting factors.

Minimizing the norm of $\vec{s}$ is a non-linear problem. One way to proceed is to use the Gauss-Newton procedure, using the Freudenstein parameters that minimize the Euclidean norm of the design error as an initial guess. The guess is modified until the *normality condition*, which is the gradient of ζ with respect to $\vec{k}$, is satisfied to a specified tolerance. For the *exact* synthesis we must have

$$\frac{\partial \zeta}{\partial \vec{k}} = \vec{0}$$

From the normality condition we have

$$\frac{\partial \zeta}{\partial \vec{k}} = \left(\frac{\partial \vec{s}}{\partial \vec{k}}\right)^T_{3\times m} \frac{\partial \zeta}{\partial \vec{s}}_{m\times 1}. \tag{3.12}$$

Lets set $\tilde{W} = \tilde{I}$. Then

$$\begin{aligned} \zeta &= \frac{1}{2}\vec{s}^T\vec{s}, \\ &= \frac{1}{2}||\vec{s}||_2^2 \;\Rightarrow\; min\; \vec{k}, \end{aligned}$$

which implies

$$\frac{\partial \zeta}{\partial \vec{s}} = \vec{s}.$$

So

$$\begin{aligned} \frac{\partial \vec{s}}{\partial \vec{k}} &= \frac{\partial}{\partial \vec{k}}(\vec{\phi}_{pres} - \vec{\phi}_{gen}), \\ \text{prescribed} \;\Rightarrow\; \text{constant} \;\Rightarrow\; & \frac{\partial \vec{\phi_{pres}}}{\partial \vec{k}} = \vec{0}, \\ \Rightarrow\; \frac{\partial \vec{s}}{\partial \vec{k}} &= -\frac{\partial \vec{\phi_{gen}}}{\partial \vec{k}}. \end{aligned}$$

3.6.1 Synthesis Equations $\tilde{S}\vec{k} - \vec{b}$:

Let

$$\vec{f} = \vec{f}(\vec{\phi}_{gen}, \vec{k}; \vec{\psi}) = \vec{0} \tag{3.13}$$

$$\vec{\psi} = \begin{bmatrix} \psi_1 \\ \cdot \\ \cdot \\ \psi_n \end{bmatrix},$$

$$\vec{f} = \begin{bmatrix} f_1 \\ \cdot \\ \cdot \\ f_n \end{bmatrix}.$$

The *total derivative* of $\vec{f}$ WRT $\vec{k}$ can be written as the sum of the partials:

$$\frac{d\vec{f}}{d\vec{k}} = \left(\frac{\partial \vec{f}}{\partial \vec{k}}\right)_{m\times 3} + \left(\frac{\partial \vec{f}}{\partial \vec{\phi}_{gen}}\right)_{m\times m} \left(\frac{\partial \vec{\phi}_{gen}}{\partial \vec{k}}\right)_{m\times 3} = \tilde{0}_{m\times 3}. \tag{3.14}$$

But, we have shown that

$$\frac{\partial \vec{\phi}_{gen}}{\partial \vec{k}} = -\frac{\partial \vec{s}}{\partial \vec{k}}.$$

We have also shown that

$$\frac{\partial \vec{f}}{\partial \vec{\phi}_{gen}} = DIAG\left(\frac{\partial f_1}{\partial \phi_{gen,1}}, ..., \frac{\partial f_m}{\partial \phi_{gen,m}}\right) \equiv \tilde{D}.$$

From Equation (3.14) we have:

$$\frac{\partial \vec{f}}{\partial \vec{\phi}_{gen}} \frac{\partial \vec{\phi}_{gen}}{\partial \vec{k}} = -\frac{\partial \vec{f}}{\partial \vec{k}}.$$

Rearranging, we get:

$$\frac{\partial \vec{\phi}_{gen}}{\partial \vec{k}} = -\tilde{D}^{-1} \frac{\partial \vec{f}}{\partial \vec{k}}. \tag{3.15}$$

Since $\vec{f}$ is linear in $\vec{k}$

$$\tilde{S} = \frac{\partial \vec{f}}{\partial \vec{k}}.$$

Finally, we have established that

$$\frac{\partial \vec{\phi}_{gen}}{\partial \vec{k}} = -\frac{\partial \vec{s}}{\partial \vec{k}}.$$

All the above relations can be used to rewrite Equation (3.15) as:

$$\frac{\partial \vec{s}}{\partial \vec{k}} = \tilde{D}^{-1} \tilde{S}. \tag{3.16}$$

Sub (3.16) into (3.11)

$$\begin{aligned} \frac{\partial \zeta}{\partial \vec{k}} &= -\left(\tilde{D}^{-1} \tilde{S}\right)^T \vec{s}, \\ &= -\tilde{S}^T \tilde{D}^{-1} \vec{s}, \\ &= \vec{0} \text{ (the normality condition)} \end{aligned}$$

The Euclidean norm of the structural error $\vec{s}$ attains a minimum value when $\tilde{D}^{-1}\vec{s}$ lies in the nullspace of $\tilde{S}^T$. From equation (3.13) we can write:

$$\vec{\phi}_{gen} = \vec{\phi}_{gen}(\vec{k})$$

But we want

$$\vec{\phi}_{gen}(\vec{k}) = \vec{\phi}_{pres}$$

Assume we have an estimate for $\vec{k}_{opt}$, which we call $\vec{k}^{\upsilon}$ obtained from the υ^{th} iteration. We can introduce a correction vector $\Delta\vec{k}$ so that we get

$$\vec{\phi}_{gen}(\vec{k}^{\upsilon} + \Delta\vec{k}) \quad = \quad \vec{\phi}_{pres}. \tag{3.17}$$

Expand the LHS of equation (3.17) in a Taylor Series:

$$\vec{\phi}_{gen}(\vec{k}^{\upsilon}) + \frac{\partial\vec{\phi}_{gen}}{\partial\vec{k}}|_{\vec{k}^{\upsilon}}\Delta\vec{k} + HOT \quad = \quad \vec{\phi}_{pres}.$$

We know that

$$\frac{\partial\vec{\phi}_{gen}}{\partial\vec{k}} \quad = \quad -\tilde{D}^{-1}\tilde{S}.$$

All this leads to:

$$\begin{aligned}\tilde{D}^{-1}\tilde{S}\Delta\vec{k} &= \vec{\phi}_{gen}(\vec{k}^{\upsilon}) - \vec{\phi}_{pres}, \\ &= -\vec{s}. \end{aligned} \tag{3.18}$$

Find $\Delta\vec{k}$ as a least squares approximation of equation (3.18) for the υ^{th} iteration. Stop when

$$||\Delta\vec{k}|| \; < \; \epsilon \; > \; 0$$

$$\Delta\vec{k} \; \rightarrow \; \vec{0} \; \Rightarrow \; \tilde{S}^T\tilde{D}^{-1}\vec{s}^{\upsilon} \; \rightarrow \; \vec{0}$$

Which is *exactly* the normality condition. Procedure converges to a minimum ζ

For the synthesis equations, we have:

$$\begin{aligned}\frac{\partial f_1}{\partial\phi_{gen,1}} &= \frac{\partial}{\partial\phi_{gen,1}}\left[k_1 + k_2\cos(\phi_{gen,1}) - k_3\cos(\psi_1) - \cos(\psi_1 - \phi_{gen,1})\right], \\ &= -k_2\sin(\phi_{gen,1}) - \sin(\psi_1 - \phi_{gen,1}), \\ D &= \begin{bmatrix} \frac{\partial f_1}{\partial\phi_{gen,1}} & 0 & 0 & ... \\ 0 & \frac{\partial f_2}{\partial\phi_{gen,2}} & 0 & ... \\ 0 & ... & ... & ... \end{bmatrix}, \\ D &= \begin{bmatrix} d_{11} & 0 & 0 & ... \\ 0 & d_{22} & 0 & ... \\ 0 & 0 & d_{33} & ... \\ ... & ... & ... & ... \end{bmatrix}, \\ D^{-1} &= \begin{bmatrix} 1/d_{11} & 0 & 0 & ... \\ 0 & 1/d_{22} & 0 & ... \\ 0 & 0 & 1/d_{33} & ... \\ ... & ... & ... & ... \end{bmatrix}. \end{aligned}$$

But we need

$$\vec{\phi}_{gen}(\vec{k}) \quad = \quad \vec{\phi}_{pres}. \tag{3.19}$$

3.6.2 Linear Least-Squares Problem

In a system of linear equations $\tilde{A}_{m\times n}\vec{x} = \vec{b}$, when there are more equations than unknowns ($\Rightarrow \;\; m > n$), and the set is said to be *over-determined*, there is, in general, no solution vector $\vec{x}$. The next best thing to an *exact* solution is the best *approximate* solution. This is the solution that comes closest to satisfying all the equations simultaneously. If *closeness* is defined in the least-squares sense, then the sum of the squares of the differences between the left- and right-hand sides of $\tilde{A}_{m\times n}\vec{x} = \vec{b}$ must be minimized. Then the over-determined linear problem becomes a (usually) solvable linear problem: the linear least-squares problem.

Conceptually, we can transform the $m \times n$ system to an $n \times n$ system by pre-multiplying both sides by $\tilde{A}^T$:

$$(\tilde{A}^T_{n\times m} A_{m\times n})\vec{x}_{n\times 1} = \tilde{A}^T_{n\times m}\vec{b}_{m\times 1}$$

These are the *normal equations* of the linear least-squares problem.

It turns out that singular value decomposition can solve the normal equations (using householder reflections) without explicitly evaluating them. It is always worth repeating that direct solution of the normal equations is usually the worst way to find least-squares solutions.

In summary, given an objective function $Z = (1/2)\vec{e}^T\vec{e}$ to minimize over $\vec{x}$, in an over-determined set of equations, we get:

The normality condition

$$\frac{\partial z}{\partial \vec{x}} = \vec{0}$$

The normal equations

$$(\tilde{A}^T\tilde{A})\vec{x} - \tilde{A}^T\vec{b} = \vec{0}$$

The Naïve Approach

$$\vec{x} = (\tilde{A}^T\tilde{A})^{-1}\tilde{A}^T\vec{b}$$

3.7 Singular Value Decomposition (SVD)

Consider the matrix equation

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

The least-squares solution to a suitably large over-determined system involves the *pseudo* of *Moore-Penrose generalized* inverse of $\tilde{A}$. Given $\tilde{A}$ and $\vec{b}$, we can approximate $\vec{x}$ in a least-squares sense as:

$$\vec{x} \;\; = \;\; [\tilde{A}^T\tilde{A}]^{-1}\tilde{A}^T\vec{b}. \tag{3.20}$$

Practically, however, *if* $\tilde{A}$ *has full rank*, then householder reflections *(HR)* are employed to transform $\tilde{A}$. They represent a change of basis that do not change the relative angles between the basis *column* vectors of $\tilde{A}$. They are proper isometries (reflections, in fact) that do not distort the metric on $\tilde{A}$. The i^{th} HR is defined such that

$$\tilde{H}_i^T \tilde{H}_i = I, \;\; \det(H_i) = -1$$

Thus, $\tilde{H}_i$ is orthogonal, but not necessarily symmetric. The dimensions of the system are shown in Figure 3.21. n HR's are required to transform $\tilde{A}$. Each HR is an $n \times m$ matrix that transforms the system into a set of n *canonical* equations as shown in Figure 3.22. Remember: No self-respecting engineer would *ever* use the pseudo inverse to approximate $\vec{x}$, but HR, when $\tilde{A}$ has full rank. What if $\tilde{A}$ does not? Enter SVD. It is *much* more than the flavor-of-the-month least-squares numerical approximation algorithm. It is a very powerful technique for dealing with sets of equations that are either singular (contain linear dependencies), numerically *close* to singular, or even fully determined and well constrained.

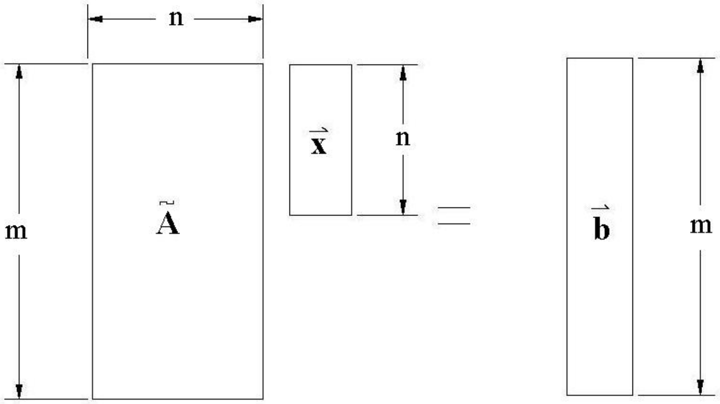


Figure 3.21:

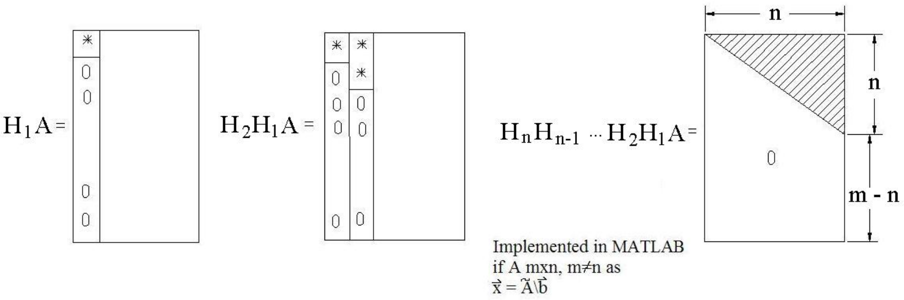


Figure 3.22:

3.7.1 Nullspace, Range, Rank:

The nullspace, range, and rank of a matrix are important concepts. consider

$$\tilde{A}\vec{x} = \vec{b}$$

Matrix $\tilde{A}$ may be viewed as a map from the vector space $\vec{x}$ to the vector space $\vec{b}$

Nullspace: The nullspace of $\tilde{A}$ is a subspace of $\vec{x}$ that $\tilde{A}$ maps to $\vec{0}$. It is the set of all linearly independent $\vec{x}$ that satisfy

$$\tilde{A}\vec{x} = \vec{0}$$

The dimension of the nullspace is called the *nullity of* $\tilde{A}$.

Range: The range of $\tilde{A}$ is the subspace of $\vec{b}$ that can be reached by $\tilde{A}$ for the $\vec{x}$ *not* in the nullspace of $\tilde{A}$

Rank: The dimension of the range is called the *rank* of $\tilde{A}$. For an $n \times n$ square matrix, $rank + nulity = n$. For a non-singular matrix A, rank $= n$.

The maximum *effective rank* of a rectangular $m \times n$ matrix is the rank of the largest square sub-matrix. For $m > n$, full rank means $rank = n$ and rank deficiency means $rank < n$

SVD is based on the fact that any (every) $m \times n$ matrix $\tilde{A}$ can be decomposed into the product of the following matrix *factors*:

$$\tilde{A}_{m\times n} = \tilde{U}_{m\times n}\tilde{\Sigma}_{n\times n}\tilde{V}^T_{n\times n}$$

$$Full\ Size:\ U_{m\times m}\Sigma_{m\times n}V^T_{n\times n}$$

Where:

- $\tilde{U}_{m\times n}$ is column-orthonormal (the column vectors are all mutually orthogonal unit vectors). In other words

$$\begin{aligned} U(:,j)\,\mathbf{.}\,U(:,k) &= 1,\ j = k \\ &= 0,\ j \neq k \end{aligned}$$

- $\tilde{\Sigma}_{n\times n}$ is a diagonal matrix whose diagonal elements are the *singular values*,σ_i of $\tilde{A}$. The singular values of $\tilde{A}$ are the square roots of the eigenvalues of $\tilde{A}^T\tilde{A}$. They are positive numbers arranged in descending order:

$$(\tilde{A}^T\tilde{A})\vec{x} = \lambda\vec{x} \Rightarrow (\tilde{A}^T\tilde{A} - \lambda\tilde{I})\vec{x} = \vec{0}$$

$$\sigma_i = \sqrt{\lambda_i}$$

- $\tilde{V}$ is a square orthonormal matrix, i.e.

$$\tilde{V}\tilde{V}^T = \tilde{I}$$

SVD is an extremely useful computational linear algebra tool because of the numerical information exposed by the decomposition. SVD explicitly constructs orthonormal bases for both the nullspace and range of every (any) matrix $\tilde{A}$

Recall that an $m \times n$ matrix $\tilde{A}$ possessing full rank means $rank = n$. In this case, all $\sigma_i, i = 1...n$, are positive, non-zero real numbers. The matrix is (analytically) non-singular. If $\tilde{A}$ is rank deficient then some of the σ_i are identically zero. The number of $\sigma_i = 0$ corresponds to the dimension of the nullspace $(n \bullet rank(\tilde{A}))$. The σ_i are arranged in $\tilde{\Sigma}$ on the diagonal in descending order: $\sigma_1 \geq \sigma_2,\ \sigma_2 \geq \sigma - 3,\ \ldots, \sigma_{n-1} \geq \sigma_n$

- The columns of $\tilde{u}$ whose same-numbered elements σ_i are non-zero are an orthogonal set of orthonormal basis vectors that span the range of A: every possible product $\tilde{A}\vec{x} = \vec{b}$ is a linear combination of these column vectors of $\vec{U}$
- The columns of $\tilde{V}$ whose same numbered elements $\sigma_i = 0$ form an orthogonal set of orthonormal basis vectors that span the nullspace of $\tilde{A}$: every possible product $\tilde{A}\vec{x} = \vec{0}$ is a linear combination of these column vectors of $\tilde{V}$. See Figure 3.23.

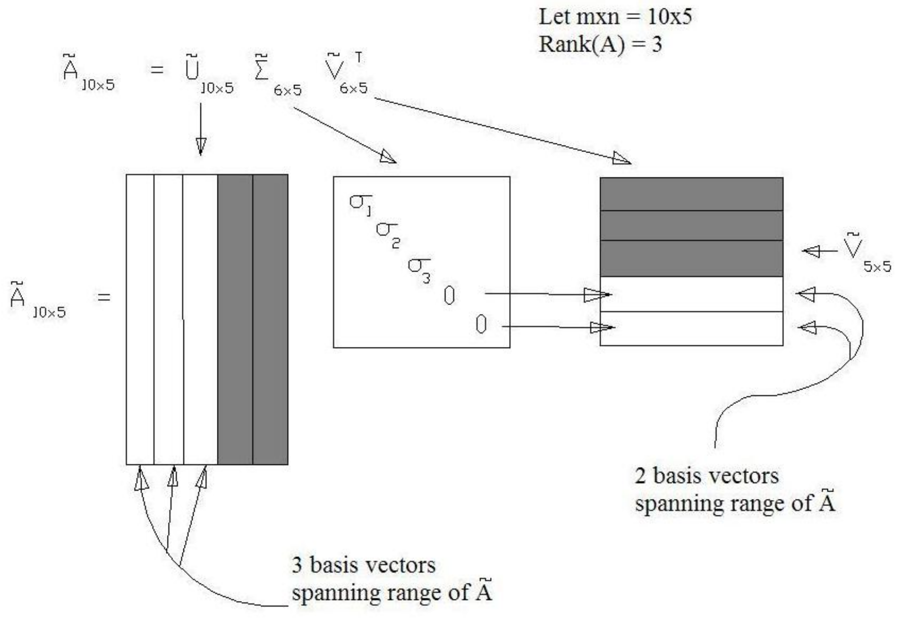


Figure 3.23:

MATLAB Note: To get the form of SVD in Figure 3.23, the syntax is

$$[U, S, V] = svd(A, 0)$$

For $\tilde{A}_{m\times n}$ returns $\tilde{U}_{m\times n}$, $\tilde{S}_{n\times n}$ $(Singular\ Values)$, $\tilde{V}_{n\times n}$ such that $\tilde{A} = \tilde{U}\tilde{S}\tilde{V}^T$

What computational advantage does this provide? Any (every) matrix, no matter how singular, can be inverted! By virtue of the properties of $\tilde{U}$,$\tilde{\Sigma}$,and $\tilde{V}$, the inverse of any (every) matrix $\tilde{A}$, square or not, is simply

$$\tilde{A}^{-1} = (\tilde{U}_{m\times n}\tilde{\Sigma}_{n\times n}\tilde{V}^T_{n\times n})^{-1} = \tilde{V}_{n\times n}\tilde{\Sigma}^{-1}_{n\times n}\tilde{U}^T_{n\times m}$$

Because $\tilde{V}$ and $\tilde{U}$ are orthogonal their inverses are equal to their transposes. Because $\tilde{\Sigma}$ is diagonal, its inverse has elements$1/\sigma_i$ on the diagonal and zeroes everywhere else. So we can write explicitly:

$$\tilde{A}^{-1}_{n\times m} = \tilde{V}_{n\times n}[diag(1/\sigma_i)]\tilde{U}^T_{n\times m}$$

Computational problems can only be caused if any $\sigma_i = 0$, or are so close to zero that the value of σ_i is dominated by roundoff error so that $1/\sigma_i$ becomes a very large, but equally inaccurate number, which pushes A^{-1} towards infinity in the wrong direction. That doesn't bode well for the solution $\vec{x} = \tilde{A}^{-1}\vec{b}$.

3.7.2 Numerical Conditioning

The condition number of a matrix is a measure of how *invertible* or how *close* to singular a matrix is. The condition number of a matrix is the ratio of the largest to the smallest singular values.

$$k(\tilde{A}) = \sigma_{max}/\sigma_{min}$$

A matrix is singular if $k = \infty$, and is ill-conditioned if k is very large. But, how big is large? Relative to what? The condition number has the following bounds

$$1 \leq k(\tilde{A}) \leq \infty$$

Instead, let's consider the reciprocal of $k(\tilde{A})$. Let this reciprocal be $\lambda(\tilde{A})$

$$\lambda(\tilde{A}) = 1/k(\tilde{A})$$

This ratio is bounded both from above and below

$$0 \leq \lambda(\tilde{A}) \leq 1$$

Now we can say $\tilde{A}$ is well-conditioned if $\lambda(\tilde{A}) \simeq 1$, and ill-conditioned if $\lambda(\tilde{A}) \simeq 0$. But, now how small is *too close* to zero. Well, this we can state with certainty. *Too small* is the computer's floating-point precision which is 10^{-12} for double precision. MATLAB uses double floating-point precision for all computations. Hence, if $\lambda(\tilde{A}) < 10^{-12}$, $\tilde{A}$ is ill=conditioned.

3.7.3 What to do when (if) $\tilde{A}$ is Singular

There are three cases to consider if $k(\tilde{A}) = \infty$.

1. If $\vec{b} = \vec{0}$, the solution for $\vec{x}$ is any column of $\tilde{V}$ whose corresponding $\sigma_i = 0$.
2. If $\vec{b} \neq 0$, but $\vec{b}$ is in the range of $\tilde{A}$. The singular set of equations does have a solution vector $\vec{x}$ In fact there are infinite solutions because any vector in the nullspace (any column of $\tilde{V}$ with corresponding $\sigma_i = 0$) can be added to $\vec{x}$ in any linear combination. See Figure 3.24.

   But hit is where things get really interesting. The $\vec{x}$ with the smallest Euclidean norm can be obtained by setting $\infty = 0$!! This, however, is the desperation mathematic is appears to be at first glance. If $\sigma_i = 0$, set $(1/\sigma_i) = 1/0 = 0$ in $\tilde{\Sigma}^{-1}$. Then, the $\vec{x}$ with the smallest 2-norm is

   $$\vec{x} \quad = \quad \tilde{V}[diag(1/\sigma_i)](\tilde{U}^T\vec{b}), \; i = 1, \; ..., \; n. \tag{3.21}$$

   Where for all $\sigma_i = 0$, $1/\sigma_i$ is set to zero. (Note, in Figure 3.25 the equation $\tilde{A}\vec{x} = \vec{d}$ is used for this case.)
3. If $\vec{b} \neq 0$ but is *not* in the range of $\tilde{A}$ (this is typical in over-determined systems of equations). In this case the set of equations are inconsistent, and no solution exists that simultaneously satisfies them. We can still use equation (3.21) to *construct* a solution vector $\vec{x}$ that is the closest approximation in a least-squares sense. In other words, SVD finds the $\vec{x}$ that minimizes the norm of the residual, r of the solution, where

   $$r \equiv ||\tilde{A}\vec{x} - \vec{b}||_2$$

   Note the happy similarity to the design error!! See Figure 3.26.

3.8 Numerical Reality

Thus far we have presumes that either a matrix *is* singular or *not*. From an analytic standpoint this is true, but not numerically. It is quite common for some σ_i to be very, very small, but not zero. Analytically the matrix is not singular, but numerically it is ill-conditioned. In this case Gaussian elimination, or the normal equations will yield a solution, but one when multiplied by $\tilde{A}$ very poorly approximates $\vec{b}$.

The solution vector $\vec{x}$ obtained by *zeroing* the small σ_i and then using equation (3.21) usually gives a better approximation than direct methods and the SVD solution where small σ_i ar not zeroed.

Small σ_i are dominated by, and hence are an artifact of, roundoff error. But again we must ask "How small is too small?". A plausible answer is to edit $(1/\sigma - i)$ if $(\sigma_i/\sigma_{max}) < \mu\epsilon$ where μ is a constant and ϵ is the machine precision. MATLAB will tell you what ϵ is on your machine: in the command window, type *eps* followed by pressing *enter*. On my desktop $\epsilon = 2.2204 \times 10^{-16}$. It

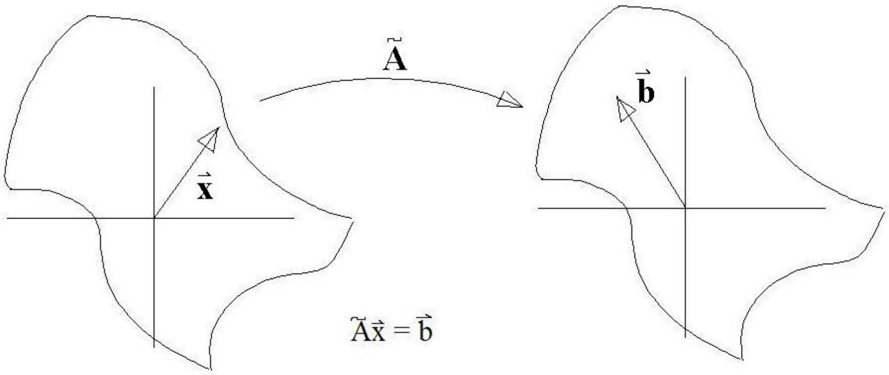


Figure 3.24: Matrix $\tilde{A}$ is non-singular. Maps one vector space onto one of the same dimension if $\tilde{A}$ is square. For the $m \times n$ case $\vec{x}$ is mapped onto $\vec{b}$ so that $\tilde{A}\vec{x} = \vec{b}$ is satisfied.

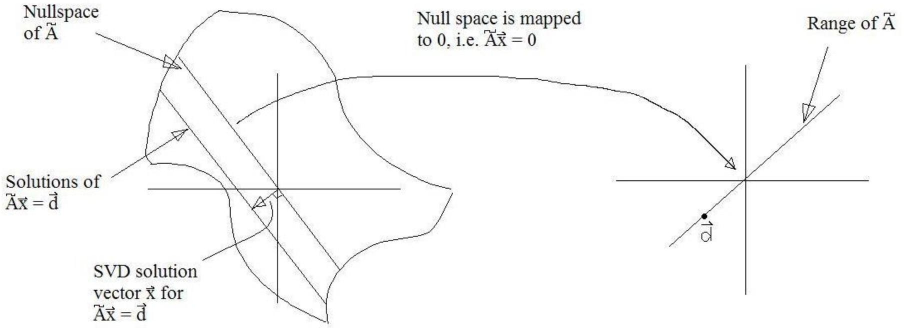

Nullspace of $\tilde{A}$
Null space is mapped to 0, i.e. $\tilde{A}\vec{x} = 0$
Range of $\tilde{A}$
Solutions of $\tilde{A}\vec{x} = \vec{d}$
SVD solution vector $\vec{x}$ for $\tilde{A}\vec{x} = \vec{d}$
$\vec{d}$

Figure 3.25: Matrix $\tilde{A}$ is singular. If $\tilde{A}$ is square it maps a vector space of lower dimension (in the figure $\tilde{A}$ maps a plane onto a line, the range of $\tilde{A}$). The nullspace of $\tilde{A}$ maps to zero. The "solution" vector $\vec{x}$ of $\tilde{A}\vec{x} = \vec{d}$ consist of a particular solution plus any vector in the nullspace. In this example, these solutions form a line parallel to the nullspace. SVD finds the $\vec{x}$ closest to zero by setting $1/\sigma_i = 0$ for all $\sigma_i = 0$. Similar statements hold for rectangular $\tilde{A}$

turns out that a useful value for μ is $rank(V)$. So we can state the following, easy rule for editing $1/\sigma_i$:

$$Set\ 1/\sigma - i = 0\ if\ (\sigma_i/\sigma_{max}) < rank(V)\epsilon$$

At first it may seem that zeroing the reciprocal of a singular value makes a bad situation worse by eliminating one linear combination of the set of equations we are attempting to solve. But we are really eliminating a linear combination of equations that is so corrupted by roundoff error that it is, at best, useless. In fact, worse than useless because it pushes the solution vector towards infinity in

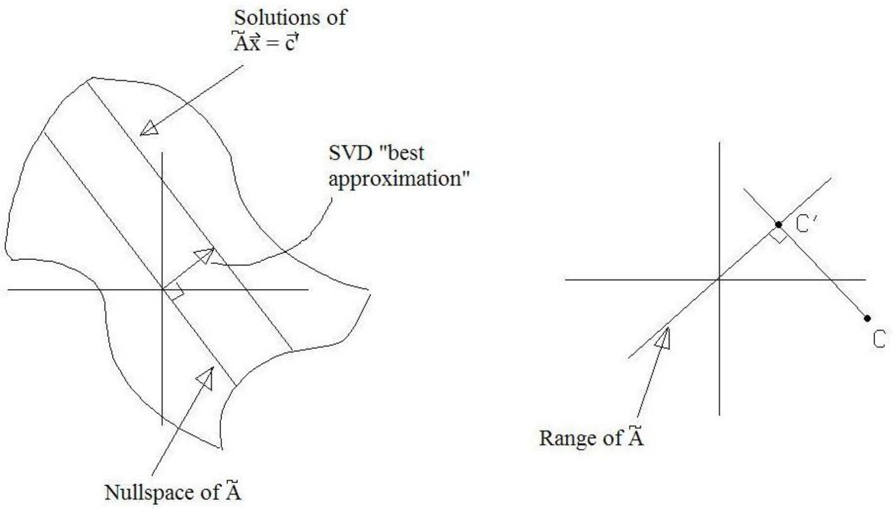


Figure 3.26: The vector $\vec{c}$ is outside the range of $\tilde{A}$, so there is no $\vec{x}$ that can satisfy $\tilde{A}\vec{x} = \vec{c}$. SVD finds the least-squares "best approximation" solution $\tilde{A}\vec{x} = \vec{c'}$, as illustrated, after all $1/\sigma_i = \infty$ have been zeroed.

a direction parallel to a nullspace vector. This makes the roundoff error worse by inflating the residual $r = ||\tilde{A}\vec{x} - \vec{b}||_2$.

In general the matrix Σ will not be singular and no σ_i will have to be eliminated, unless there are column degeneracies in $\tilde{A}$ (near linear combinations of the columns). The i^{th} column in $\tilde{V}$ corresponding to the annihilated σ_i gives the linear combination of the elements of $\vec{x}$ that are ill-determined even by the over-determined set of equations. These elements of $\vec{x}$ are insensitive to the data, and should not be trusted.