ME751
Advanced Computational Multibody Dynamics

Section 9.6

February 25, 2010

"In China if you are one in a million – there are 1,300 other people just like you."
Bill Gates

Before we get started…

- Last Time:
  - Finished Geometric Constraints
  - Started discussion about their partial derivatives

- Today:
  - Discussion of assignment
  - Finish partial derivatives
  - Discuss computation of $\bar{\mathbf{\Pi}}$
  - Quick remarks on Position Analysis + Newton Raphson
  - Start the dynamics problem

- HW6 – due on March 4, posted online today
  - Heads up: next week's HW7 will ask you to define your ME751 Final Project

- Student Feedback: posted online on class website
  - Only two students provided feedback

Today's Assigned HW

- First in a series of assignments that each contains a MATLAB component
  - For first one you'll have to implement computational support for two basic GCon's
- You will be coding over the next four or five HWs a series of MATLAB functions that in the end should form a cohesive 3D Simulation Engine for Kinematics and Dynamics Analysis
- On Th, March 4, Hammad will give a 15 minute presentation of the steps you need to take in order to visualize the motion of your model
  - Using Ogre, an open source 3D Graphics Engine (http://www.ogre3d.org/)
- The task of putting together 3D Simulation Engine can become your Final Project

[1/3]
3D Simulation Engine: Comments

- At least three A grades will be assigned in conjunction with 3D Simulation Engine:
  - Category Speed: The student with the fastest code on a benchmark problem gets an automatic A grade in ME751
  - Category Functionality: The student with the implementation with the richest functionality will get an automatic A grade
    - Functionality: defined as the number of GCon's supported, analysis modes implemented (Kinematics, Dynamics, Equilibrium, Inverse Dynamics, etc.), integration method supported, etc.
  - Category Pre-Processing: The student with the best way to define a multi-body model (preferably GUI-based model definition) will get an automatic A grade
    - Might or might not be offered
  - Category Post-Processing: The student with the best way to visualize simulation results will get an automatic A grade

[2/3]

3D Simulation Engine: Comments

- The code of each of you will be made available online at the course website (under the "Students' Code" link)
  - Privacy concerns: we'll use code names to protect your identity
    - Please send me your code name (Examples: jamesbond, batman, robinhood, etc.)
      - If you don't provide a code name I'll assign one and email to you (you don't want this to happen)
      - Use the following email message "ME751 code name: jamesbond". Don't include anything in the body of the email since I'm not going to read it

- The TA will not debug your code. He'll only try to validate it using a set of different input data

- If you did mess up the code for week 2 (you'll know if this is the case once you get your HW score back), when working on week 3 assignment it might be wise to look on the class website at other people's code and recycle it
  - NOTE: For this approach to work, I will not accept late HW

[3/3]

3D Simulation Engine: Comments

- When you email your code; that is, when you submit your homework, zip all your MATLAB files and email to me
  - Include *all* your MATLAB files; i.e., Week 3 will include your week 2 and week 1 files, etc.
  - Use the following email subject: “ME751: 3D Simulation Engine” and don’t include anything *except* a zip file. Any questions you might have, email them separately

- The way the TA will check your code will be by running a MATLAB file called “simEngine3D.mat”
  - In other words, make sure you run your code by executing this MATLAB file, which in turn calls other MATLAB files that you implement
  - Why? Because we don’t want to learn how to run the program of each and every one of you. This will enforce a unique entry point to all 3D Simulation Engine developed in ME751

- This is a multi-week project – please make sure you USE COMMENTS heavily in your code. Otherwise, in week 4 you’ll struggle to understand what you did in week 1

Basic GCon DP1: $\mathbf{\Phi}_{\mathbf{r}}^{DP1}$ and $\mathbf{\Phi}_{\mathbf{p}}^{DP1}$

- Recall that

$$\Phi^{DP1}(i, \bar{\mathbf{a}}_i, j, \bar{\mathbf{a}}_j, f(t)) = \bar{\mathbf{a}}_i^T \mathbf{A}_i^T \mathbf{A}_j \bar{\mathbf{a}}_j - f(t) = \mathbf{a}_i{}^T \mathbf{a}_j - f(t) = 0$$

- Then, it follows that

$$\frac{\partial \Phi^{DP1}}{\partial \mathbf{r}_i} = \mathbf{0}_{1\times 3} \qquad \frac{\partial \Phi^{DP1}}{\partial \mathbf{p}_i} = \mathbf{a}_j^T \mathbf{B}\left(\mathbf{p}_i, \bar{\mathbf{a}}_i\right)$$

$$\frac{\partial \Phi^{DP1}}{\partial \mathbf{r}_j} = \mathbf{0}_{1\times 3} \qquad \frac{\partial \Phi^{DP1}}{\partial \mathbf{p}_j} = \mathbf{a}_i^T \mathbf{B}\left(\mathbf{p}_j, \bar{\mathbf{a}}_j\right)$$

- Putting it all together (note that $\Phi_{\mathbf{q}}^{DP1} \in \mathbb{R}^{1\times 7nb}$) ,

$$\Phi_{\mathbf{q}}^{DP1} = \left[\mathbf{0}_{1\times 3} \ldots \mathbf{0}_{1\times 3} \ldots \mathbf{0}_{1\times 3} \ldots \ldots \mathbf{0}_{1\times 3} \ \frac{\partial \Phi^{DP1}}{\partial \mathbf{p}_i} \ \mathbf{0}_{1\times 3} \ldots \mathbf{0}_{1\times 3} \ \frac{\partial \Phi^{DP1}}{\partial \mathbf{p}_j} \ \mathbf{0}_{1\times 3} \ldots \mathbf{0}_{1\times 3}\right]$$

Partials with respect to $\mathbf{r}$

Partials with respect to $\mathbf{p}$

$$= \left[\mathbf{0}_{1\times 3} \ldots \mathbf{0}_{1\times 3} \ldots \mathbf{0}_{1\times 3} \ldots \ldots \mathbf{0}_{1\times 3} \ \mathbf{a}_j^T \mathbf{B}\left(\mathbf{p}_i, \bar{\mathbf{a}}_i\right) \ \mathbf{0}_{1\times 3} \ldots \mathbf{0}_{1\times 3} \ \mathbf{a}_i^T \mathbf{B}\left(\mathbf{p}_j, \bar{\mathbf{a}}_j\right) \ \mathbf{0}_{1\times 3} \ldots \mathbf{0}_{1\times 3}\right]$$

Body 1, **r** | Body i, **r** | Body j, **r** | Body i-1, **p** | Body i, **p** | Body i+1, **p** | Body j-1, **p** | Body j, **p** | Body j+1, **p** | Body nb, **p**

[Short Detour]:
Computing $[\mathbf{d}_{ij}]_{\mathbf{q}}$

- Recall that

$$\mathbf{d}_{ij} = \mathbf{r}_j + \mathbf{A}_j \bar{\mathbf{s}}_j^Q - \mathbf{r}_i - \mathbf{A}_i \bar{\mathbf{s}}_i^P = \mathbf{r}_j + \mathbf{s}_j^Q - \mathbf{r}_i - \mathbf{s}_i^P$$

- It follows that

$$\begin{aligned}
[\mathbf{d}_{ij}]_{\mathbf{q}_i,\mathbf{q}_j} &= [-\mathbf{I}_3 \quad -(\mathbf{s}_i^P)_{\mathbf{p}_i} \quad \mathbf{I}_3 \quad (\mathbf{s}_j^Q)_{\mathbf{p}_j}] \\
&= [-\mathbf{I}_3 \quad -\mathbf{B}(\mathbf{p}_i, \bar{\mathbf{s}}_i^P) \quad \mathbf{I}_3 \quad \mathbf{B}(\mathbf{p}_j, \bar{\mathbf{s}}_j^Q)]
\end{aligned}$$

Basic GCon DP2: $\mathbf{\Phi}_{\mathbf{r}}^{DP2}$ and $\mathbf{\Phi}_{\mathbf{p}}^{DP1}$

- Recall that

$$\Phi^{DP2}(i, \bar{\mathbf{a}}_i, \bar{\mathbf{s}}_i^P, j, \bar{\mathbf{s}}_j^Q, f(t)) = \bar{\mathbf{a}}_i^T \mathbf{A}_i^T \mathbf{d}_{ij} - f(t) = \mathbf{a}_i{}^T \mathbf{d}_{ij} - f(t) = 0$$

- It follows that

$$\begin{aligned}
\Phi^{DP2}_{\mathbf{q}_i,\mathbf{q}_j}(\mathbf{a}_i, \mathbf{d}_{ij}) &= \mathbf{a}_i^T (\mathbf{d}_{ij})_{\mathbf{q}_i,\mathbf{q}_j} + \mathbf{d}_{ij}^T (\mathbf{a}_i)_{\mathbf{q}_i,\mathbf{q}_j} \\
&= \mathbf{a}_i^T \begin{bmatrix} -\mathbf{I}_3 & -\mathbf{B}(\mathbf{p}_i, \bar{\mathbf{s}}_i^P) & \mathbf{I}_3 & \mathbf{B}(\mathbf{p}_j, \bar{\mathbf{s}}_j^Q) \end{bmatrix} + \mathbf{d}_{ij}^T \begin{bmatrix} \mathbf{0} & \mathbf{B}(\mathbf{p}_i, \bar{\mathbf{a}}_i) & \mathbf{0} & \mathbf{0} \end{bmatrix} \\
&= \begin{bmatrix} -\mathbf{a}_i^T & \mathbf{d}_{ij}^T \mathbf{B}(\mathbf{p}_i, \bar{\mathbf{s}}_i^P) - \mathbf{a}_i^T \mathbf{B}(\mathbf{p}_i, \bar{\mathbf{s}}_i^P) & \mathbf{a}_i^T & \mathbf{a}_i^T \mathbf{B}(\mathbf{p}_j, \bar{\mathbf{s}}_j^Q) \end{bmatrix}
\end{aligned}$$

Basic GCon D: $\mathbf{\Phi}_{\mathbf{r}}^{D}$ and $\mathbf{\Phi}_{\mathbf{p}}^{D}$

- Recall that the GCon-D assumes the expression

$$\Phi^D(i, \bar{\mathbf{s}}_i^P, j, \bar{\mathbf{s}}_j^Q, f(t)) = \mathbf{d}_{ij}^T \mathbf{d}_{ij} - f(t) = 0$$

- It follows that

$$\begin{aligned}
\Phi^D_{\mathbf{q}_i,\mathbf{q}_j} &= (\mathbf{d}_{ij}^T \mathbf{d}_{ij})_{\mathbf{q}_i,\mathbf{q}_j} = 2\mathbf{d}_{ij}^T [\mathbf{d}_{ij}]_{\mathbf{q}_i,\mathbf{q}_j} \\
&= 2\mathbf{d}_{ij}^T [\ -\mathbf{I}_3 \quad -\mathbf{B}(\mathbf{p}_i, \bar{\mathbf{s}}_i^P) \quad \mathbf{I}_3 \quad \mathbf{B}(\mathbf{p}_j, \bar{\mathbf{s}}_j^Q)\ ] \\
&= [\ -2\mathbf{d}_{ij}^T \quad -2\mathbf{d}_{ij}^T \mathbf{B}(\mathbf{p}_i, \bar{\mathbf{s}}_i^P) \quad 2\mathbf{d}_{ij}^T \quad 2\mathbf{d}_{ij}^T \mathbf{B}(\mathbf{p}_j, \bar{\mathbf{s}}_j^Q)\ ]
\end{aligned}$$

Basic GCon CD: $\mathbf{\Phi}_{\mathbf{r}}^{CD}$ and $\mathbf{\Phi}_{\mathbf{p}}^{CD}$

- Recall that the GCon-CD assumes the expression

$$\Phi^{CD}(\mathbf{c}, i, \bar{\mathbf{s}}_i^P, j, \bar{\mathbf{s}}_j^Q, f(t)) = \mathbf{c}^T \mathbf{d}_{ij} - f(t) = 0$$

- It follows that

$$\begin{aligned}
\Phi^D_{\mathbf{q}_i,\mathbf{q}_j} &= (\mathbf{c}^T \mathbf{d}_{ij})_{\mathbf{q}_i,\mathbf{q}_j} = \mathbf{c}^T [\mathbf{d}_{ij}]_{\mathbf{q}_i,\mathbf{q}_j} \\
&= \mathbf{c}^T \begin{bmatrix} -\mathbf{I}_3 & -\mathbf{B}(\mathbf{p}_i, \bar{\mathbf{s}}_i^P) & \mathbf{I}_3 & \mathbf{B}(\mathbf{p}_j, \bar{\mathbf{s}}_j^Q) \end{bmatrix} \\
&= \begin{bmatrix} -\mathbf{c}^T & -\mathbf{c}^T \mathbf{B}(\mathbf{p}_i, \bar{\mathbf{s}}_i^P) & \mathbf{c}^T & \mathbf{c}^T \mathbf{B}(\mathbf{p}_j, \bar{\mathbf{s}}_j^Q) \end{bmatrix}
\end{aligned}$$

End Computing Partial Derivatives $\mathbf{\Phi_r}$ and $\mathbf{\Phi_p}$

Start Computing $\bar{\mathbf{\Pi}}$.

Computing $\bar{\mathbf{\Pi}}$

- Recall that in last lecture we commented on some notation used in Haug's book (table 9.4.1, pp. 357; Professor Haug provided last week a handout that addresses these issues and I will post on the class website under link "Chapter 9 Supplement"). We concluded that:
  - *By definition*, $\bar{\mathbf{\Pi}}$ is the coefficient matrix that multiplies $\bar{\omega}$ in the expression of the time derivative $\dot{\mathbf{\Phi}}(\mathbf{q}, t)$
  - The $\mathbf{\Phi}_{\bar{\pi}}$ of the book is denoted in this class by $\bar{\mathbf{\Pi}}$
  - $\bar{\pi}$ is not a variable
  - $\mathbf{\Phi}_{\bar{\pi}}$ is not a partial derivative
- The matrix $\bar{\mathbf{R}} = \left[\mathbf{\Phi}_{\mathbf{r}} \;\; \bar{\mathbf{\Pi}}\right]$ was introduced and used to get the velocities $\dot{\mathbf{r}}$ and $\bar{\omega}$ (instead of $\dot{\mathbf{r}}$ and $\dot{\mathbf{p}}$):

$$\frac{\mathrm{d}\mathbf{\Phi}(\mathbf{r}, \mathbf{p}, t)}{\mathrm{d}t} = \mathbf{\Phi}_{\mathbf{r}}\dot{\mathbf{r}} + \bar{\mathbf{\Pi}}\bar{\omega} + \mathbf{\Phi}_t = [\mathbf{\Phi}_{\mathbf{r}} \;\; \bar{\mathbf{\Pi}}] \begin{bmatrix} \dot{\mathbf{r}} \\ \bar{\omega} \end{bmatrix} + \mathbf{\Phi}_t = \mathbf{0}_{6nb} \Rightarrow \bar{\mathbf{R}} \begin{bmatrix} \dot{\mathbf{r}} \\ \bar{\omega} \end{bmatrix} = \nu_{6nb}$$

- Note that a similar matrix $\mathbf{R} = [\mathbf{\Phi}_{\mathbf{r}} \;\; \mathbf{\Pi}]$ can be introduced, used to compute $\dot{\mathbf{r}}$ and ω:

$$\frac{\mathrm{d}\mathbf{\Phi}(\mathbf{r}, \mathbf{p}, t)}{\mathrm{d}t} = \mathbf{\Phi}_{\mathbf{r}}\dot{\mathbf{r}} + \mathbf{\Pi}\omega + \mathbf{\Phi}_t = [\mathbf{\Phi}_{\mathbf{r}} \;\; \mathbf{\Pi}] \begin{bmatrix} \dot{\mathbf{r}} \\ \omega \end{bmatrix} + \mathbf{\Phi}_t = \mathbf{0}_{6nb} \Rightarrow \mathbf{R} \begin{bmatrix} \dot{\mathbf{r}} \\ \omega \end{bmatrix} = \nu_{6nb}$$

Basic GCon DP1: $\bar{\mathbf{\Pi}}_i^{DP1}$ and $\bar{\mathbf{\Pi}}_j^{DP1}$

- Recall that

$$\Phi^{DP1}(i, \bar{\mathbf{a}}_i, j, \bar{\mathbf{a}}_j, f(t)) = \bar{\mathbf{a}}_i^T \mathbf{A}_i^T \mathbf{A}_j \bar{\mathbf{a}}_j - f(t) = \mathbf{a}_i{}^T \mathbf{a}_j - f(t) = 0$$

- Then, it follows that

$$\begin{aligned}
\dot{\Phi}^{DP1}(i, \bar{\mathbf{a}}_i, j, \bar{\mathbf{a}}_j, f(t)) &= \bar{\mathbf{a}}_i^T \mathbf{A}_i^T \dot{\mathbf{A}}_j \bar{\mathbf{a}}_j + \bar{\mathbf{a}}_j^T \mathbf{A}_j^T \dot{\mathbf{A}}_i \bar{\mathbf{a}}_i - \dot{f}(t) \\
&= \bar{\mathbf{a}}_i^T \mathbf{A}_i^T \mathbf{A}_j \tilde{\bar{\omega}}_j \bar{\mathbf{a}}_j + \bar{\mathbf{a}}_j^T \mathbf{A}_j^T \mathbf{A}_i \tilde{\bar{\omega}}_i \bar{\mathbf{a}}_i - \dot{f}(t) \\
&= \boxed{-\bar{\mathbf{a}}_i^T \mathbf{A}_i^T \mathbf{A}_j \tilde{\bar{\mathbf{a}}}_j} \bar{\omega}_j \boxed{-\bar{\mathbf{a}}_j^T \mathbf{A}_j^T \mathbf{A}_i \tilde{\bar{\mathbf{a}}}_i} \bar{\omega}_i - \dot{f}(t)
\end{aligned}$$

- Therefore, we end up with

$$\bar{\mathbf{\Pi}}_i^{DP1} = -\bar{\mathbf{a}}_j^T \mathbf{A}_j^T \mathbf{A}_i \tilde{\bar{\mathbf{a}}}_i \qquad \text{AND} \qquad \bar{\mathbf{\Pi}}_j^{DP1} = -\bar{\mathbf{a}}_i^T \mathbf{A}_i^T \mathbf{A}_j \tilde{\bar{\mathbf{a}}}_j$$

Basic GCon DP2: $\bar{\mathbf{\Pi}}_i^{DP2}$ and $\bar{\mathbf{\Pi}}_j^{DP2}$

- Recall that

$$\Phi^{DP2}(i, \bar{\mathbf{a}}_i, \bar{\mathbf{s}}_i^P, j, \bar{\mathbf{s}}_j^Q, f(t)) = \bar{\mathbf{a}}_i^T \mathbf{A}_i^T \mathbf{d}_{ij} - f(t) = \mathbf{a}_i{}^T \mathbf{d}_{ij} - f(t) = 0$$

- It follows that

$$\begin{aligned} \dot{\Phi}^{DP2}(\mathbf{a}_i, \mathbf{d}_{ij}) &= \mathbf{a}_i^T \dot{\mathbf{d}}_{ij} + \mathbf{d}_{ij}^T \dot{\mathbf{a}}_i - \dot{f}(t) \\ &= \ldots \end{aligned}$$

- Just like before, move $\bar{\omega}_i$ and $\bar{\omega}_j$ at the right end of the terms in which they show up to obtain that

$$\bar{\mathbf{\Pi}}_i^{DP2} = \bar{\mathbf{a}}_i^T \tilde{\bar{\mathbf{s}}}_i^P - \mathbf{d}_{ij}^T \mathbf{A}_i \tilde{\bar{\mathbf{a}}}_i \qquad \text{AND} \qquad \bar{\mathbf{\Pi}}_j^{DP2} = -\bar{\mathbf{a}}_i^T \mathbf{A}_i^T \mathbf{A}_j \tilde{\bar{\mathbf{s}}}_j^Q$$

Basic GCon D: $\bar{\mathbf{\Pi}}_i^D$ and $\bar{\mathbf{\Pi}}_j^D$

- Recall that the GCon-D assumes the expression

$$\Phi^D(i, \bar{\mathbf{s}}_i^P, j, \bar{\mathbf{s}}_j^Q, f(t)) = \mathbf{d}_{ij}^T \mathbf{d}_{ij} - f(t) = 0$$

- It follows that

$$\begin{aligned} \dot{\Phi}^D &= 2\mathbf{d}_{ij}^T \dot{\mathbf{d}}_{ij} - \dot{f}(t) \\ &= \ldots \end{aligned}$$

- Just like before, move $\bar{\omega}_i$ and $\bar{\omega}_j$ at the right end of the terms in which they show up to obtain that

$$\bar{\mathbf{\Pi}}_i^D = 2\mathbf{d}_{ij}^T \mathbf{A}_i \tilde{\bar{\mathbf{s}}}_i^P \qquad \text{AND} \qquad \bar{\mathbf{\Pi}}_j^D = -2\mathbf{d}_{ij}^T \mathbf{A}_j \tilde{\bar{\mathbf{s}}}_j^Q$$

Basic GCon CD: $\bar{\mathbf{\Pi}}_i^{CD}$ and $\bar{\mathbf{\Pi}}_j^{CD}$

- Recall that the GCon-D assumes the expression

$$\Phi^{CD}(\mathbf{c}, i, \bar{\mathbf{s}}_i^P, j, \bar{\mathbf{s}}_j^Q, f(t)) = \mathbf{c}^T \mathbf{d}_{ij} - f(t) = 0$$

- It follows that

$$\begin{aligned} \dot{\Phi}^D &= \mathbf{c}^T \dot{\mathbf{d}}_{ij} - \dot{f}(t) \\ \\ &= \ldots \end{aligned}$$

- Just like before, move $\bar{\omega}_i$ and $\bar{\omega}_j$ at the right end of the terms in which they show up to obtain that

$$\bar{\mathbf{\Pi}}_i^D = \mathbf{c}^T \mathbf{A}_i \tilde{\bar{\mathbf{s}}}_i^P \qquad \text{AND} \qquad \bar{\mathbf{\Pi}}_j^D = -\mathbf{c}^T \mathbf{A}_j \tilde{\bar{\mathbf{s}}}_j^Q$$

Final Comments on the Content of $\mathbf{\Phi}(\mathbf{q}, t)$

- These final thoughts are motivated by the fact that when dealing with Euler Parameters we need to clarify what exactly we mean by $\mathbf{\Phi}(\mathbf{q}, t)$

- Up to this point, we said that

$$\mathbf{\Phi}(\mathbf{q}, t) = \begin{bmatrix} \mathbf{\Phi}^K(\mathbf{q}) \\ \\ \mathbf{\Phi}^D(\mathbf{q}, t) \end{bmatrix}$$

- It turns out that this is painting a picture with too wide of a brush. Specifically, since I work with Euler Parameters I will also have to explicitly include in (q,t) the Euler Parameter normalization constraints

- We will make the following distinction from now on (two notation conventions):

  – We will denote by $\mathbf{\Phi}^{\mathbf{P}}$ the specific set of nb Euler Parameter normalization constraints:

$$\mathbf{\Phi}^{\mathbf{P}}(\mathbf{q}) = \mathbf{\Phi}^{\mathbf{P}}(\mathbf{r}, \mathbf{p}) = \mathbf{\Phi}^{\mathbf{P}}(\mathbf{p}) = \begin{bmatrix} \mathbf{p}_1^T \mathbf{p}_1 - 1.0 \\ \cdots \\ \mathbf{p}_{nb}^T \, \mathbf{p}_{nb} - 1.0 \end{bmatrix}$$

  – We will denote by $\mathbf{\Phi}^K(\mathbf{q})$ the set of ACE that are associated with a GCon present in the system

Final Comments on the Content of $\mathbf{\Phi}(\mathbf{q}, t)$

- Subsequently, when carrying out Velocity Analysis, if solving for $\dot{\mathbf{r}}$ and $\dot{\mathbf{p}}$, you have to include $\mathbf{\Phi}^{\mathbf{p}} = \mathbf{0}_{nb}$ in the set of constraints $\mathbf{\Phi}(\mathbf{q}, t)$
  - Justification: although not stemming from a GCon, the Euler Parameter normalization constraints are nonetheless constraints induced by the particular choice of generalized coordinates. It's important to make this distinction between ACEs (a) stemming from the geometry of the motion (from GCon's), and (b) induced by the choice of generalized coordinates we've decided to work with (normalization constraints for $\mathbf{p}$, in our case).

- Note that for the Velocity and Acceleration right-hand side of the linear equations, you have:

$$\nu^{\mathbf{p}} = \mathbf{0}_{nb} \qquad \& \qquad \gamma^{\mathbf{p}} = \begin{bmatrix} -2\dot{\mathbf{p}}_1^T \dot{\mathbf{p}}_1 \\ \cdots \\ -2\dot{\mathbf{p}}_{nb}^T \dot{\mathbf{p}}_{nb} \end{bmatrix}$$

- Notice that you don't have to be concerned with the Euler Parameter normalization constraints (their time derivative, that is) if you solve for $\dot{\mathbf{r}}$ and $\bar{\omega}$. Since you compute $\dot{\mathbf{p}}$ as $\dot{\mathbf{p}} = \frac{1}{2}\mathbf{G}^T\bar{\omega}$, the velocity constraints associated with the Euler Parameter normalization constraints are automatically satisfied (Problem 2 of today's Homework).

Final Comments on the Content of $\mathbf{\Phi}(\mathbf{q}, t)$

- In order to clarify what we mean by $\mathbf{\Phi}$, we'll use the following notation (superscript F stands for 'Full')

$$\mathbf{\Phi}(\mathbf{q}, t) = \begin{bmatrix} \mathbf{\Phi}^K(\mathbf{q}) \\ \mathbf{\Phi}^D(\mathbf{q}, t) \end{bmatrix} = \mathbf{0}_{6nb} \qquad \mathbf{\Phi}^F(\mathbf{q}, t) = \begin{bmatrix} \mathbf{\Phi}^K(\mathbf{q}) \\ \mathbf{\Phi}^D(\mathbf{q}, t) \\ \mathbf{\Phi}^{\mathbf{P}}(\mathbf{p}) \end{bmatrix} = \mathbf{0}_{7nb}$$

- In Kinematics, when carrying out Position Analysis, one must **always** solve $\mathbf{\Phi}^F(\mathbf{q}, t) = \mathbf{0}_{7nb}$; i.e., a nonlinear system of dimension $7nb$ (as long as we use Euler Parameters as generalized coordinates).

- When carrying out Velocity Analysis, one must solve either $\mathbf{\Phi}^F_{\mathbf{q}} \dot{\mathbf{q}} = \nu^F$ (of dimension $7nb$), or $\bar{\mathbf{R}} \begin{bmatrix} \dot{\mathbf{r}} \\ \bar{\omega} \end{bmatrix} = \nu$ (of dimension $6nb$), or $\mathbf{R} \begin{bmatrix} \dot{\mathbf{r}} \\ \omega \end{bmatrix} = \nu$ (of dimension $6nb$).

- When carrying out Acceleration Analysis, one must solve either $\mathbf{\Phi}^F_{\mathbf{q}} \ddot{\mathbf{q}} = \gamma^F$ (of dimension 7nb), or $\bar{\mathbf{R}} \begin{bmatrix} \ddot{\mathbf{r}} \\ \dot{\bar{\omega}} \end{bmatrix} = \gamma$ (of dimension $6nb$), or $\mathbf{R} \begin{bmatrix} \ddot{\mathbf{r}} \\ \dot{\omega} \end{bmatrix} = \gamma$ (of dimension $6nb$).

Solving a Nonlinear System

- The most important numerical algorithm to understand in Kinematics
- Relied upon heavily by ADAMS, used almost in all analysis modes
    - Kinematics
    - Dynamics
    - Equilibrium

- How does one go about finding the solution?

$$\sqrt{x} - \sin x = 0$$

$$\begin{cases} x - e^{y} & = & 1 \\ \ln(1+x) - \cos y & = & 0 \end{cases}$$

Newton-Raphson Method

- Framework, for the one dimensional case:
  - A function $f(x)$ is given, $f : \mathbb{R} \to \mathbb{R}$. You are interested in finding the root α of the function f, or in other words, finding α that verifies the equation:

$$f(x) = 0$$

  - The assumption is that f is twice continuously differentiable

- The Newton-Raphson algorithm is an iterative algorithm that is implemented as follows:

$$x^{(1)} = x^{(0)} - \frac{f(x^{(0)})}{f'(x^{(0)})}$$

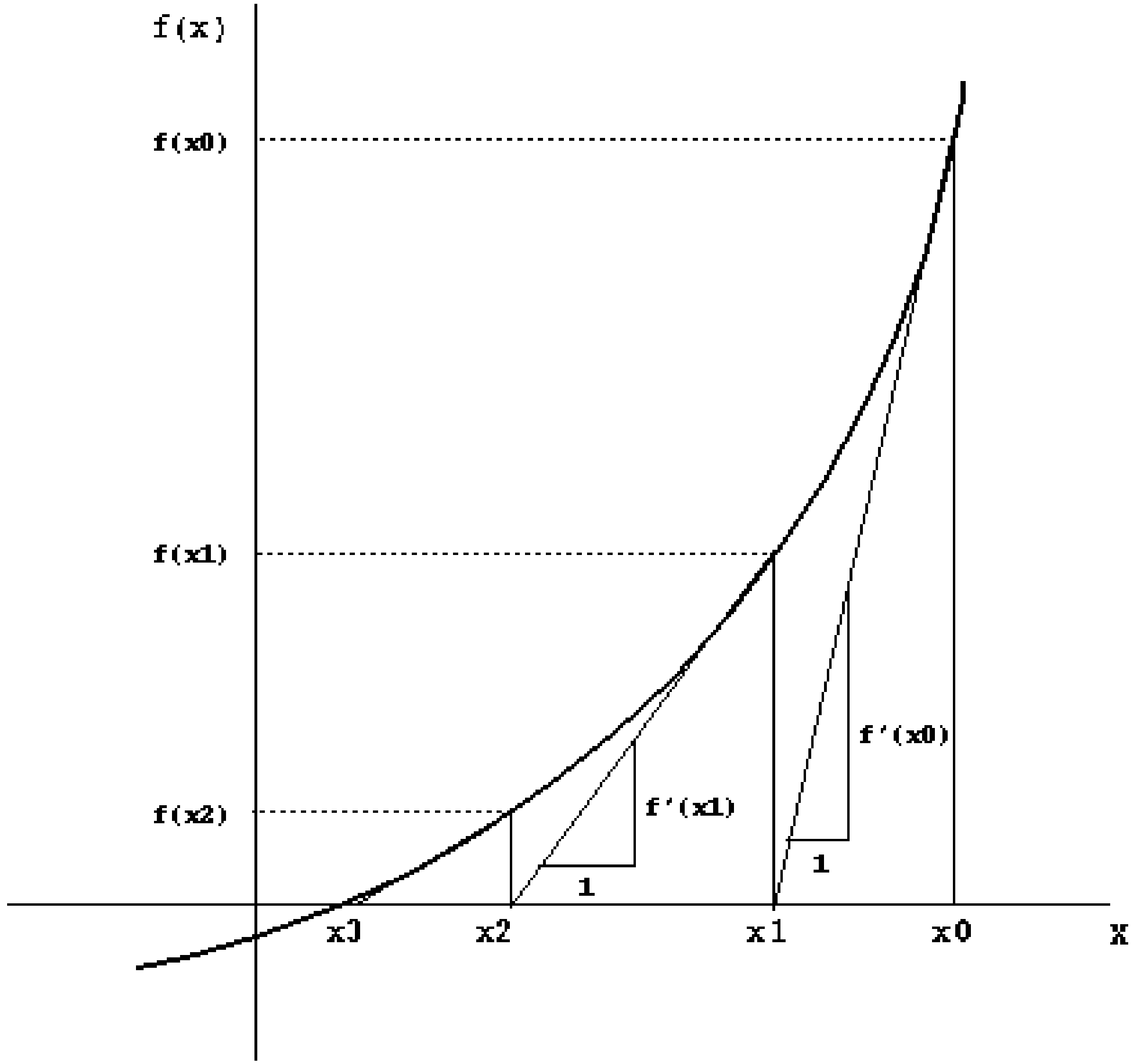
$$x^{(2)} = x^{(1)} - \frac{f(x^{(1)})}{f'(x^{(1)})}$$

$$x^{(3)} = x^{(2)} - \frac{f(x^{(2)})}{f'(x^{(2)})}$$

$$\ldots$$

  - Note that an initial guess $x^{(0)}$ is needed.
  - The iterative algorithms is stopped after taking a sufficient number of iterations that gradually get $x^{(k)}$ closer to the value α (this if nothing goes wrong...)

Newton-Raphson Method
Geometric Interpretation



Newton-Raphson
The Multidimensional Case

- Solve for $\mathbf{q} \in \mathbb{R}^{7nb}$

the nonlinear system

$$\mathbf{\Phi}^F(\mathbf{q}, t) = \begin{bmatrix} \mathbf{\Phi}^K(\mathbf{q}) \\ \mathbf{\Phi}^D(\mathbf{q}, t) \\ \mathbf{\Phi}^{\mathbf{P}}(\mathbf{p}) \end{bmatrix} = \mathbf{0}_{7nb}$$

- The algorithm becomes

$$\mathbf{q}^{(1)} = \mathbf{q}^{(0)} - \left[\mathbf{\Phi}_{\mathbf{q}}(\mathbf{q}^{(0)})\right]^{-1} \mathbf{\Phi}(\mathbf{q}^{(0)}, t)$$

- The Jacobian is defined as

$$\mathbf{\Phi}_{\mathbf{q}}(\mathbf{q}^{(0)}) = \left.\frac{\partial \mathbf{\Phi}}{\partial \mathbf{q}}\right|_{\mathbf{q}=\mathbf{q}^{(0)}}$$

Putting things in perspective…

- Newton algorithm for nonlinear systems requires:
  - A starting point $\mathbf{q}^{(0)}$ from where the solution starts being searched for
  - An iterative process in which the approximation of the solution is gradually improved:

$$\mathbf{q}^{(1)} = \mathbf{q}^{(0)} - \left[\mathbf{\Phi}_{\mathbf{q}}(\mathbf{q}^{(0)})\right]^{-1} \mathbf{\Phi}(\mathbf{q}^{(0)}, t)$$

$$\mathbf{q}^{(2)} = \mathbf{q}^{(1)} - \left[\mathbf{\Phi}_{\mathbf{q}}(\mathbf{q}^{(1)})\right]^{-1} \mathbf{\Phi}(\mathbf{q}^{(1)}, t)$$

$$\mathbf{q}^{(3)} = \mathbf{q}^{(2)} - \left[\mathbf{\Phi}_{\mathbf{q}}(\mathbf{q}^{(2)})\right]^{-1} \mathbf{\Phi}(\mathbf{q}^{(2)}, t)$$

$$\cdots \text{etc.}$$

Newton's Method: Closing Remarks

- Can ever things go wrong with Newton's method?

- Yes, there are at least three instances:

  1. Most commonly, the starting point is not close to the solution that you try to find and the iterative algorithm diverges (goes to infinity)

  2. Since a nonlinear system can have multiple solutions, the Newton algorithm finds a solution that is not the one sought (happens if you don't choose the starting point right)

  3. The speed of convergence of the algorithm is not good (happens if the Jacobian is close to being singular (zero determinant) at the root, not that common)

Newton's Method: Closing Remarks

- What can you do address these issues?
- You cannot do anything about 3 above, but can fix 1 and 2 provided you choose your starting point carefully
- Newton's method converges very fast (quadratically) if started close enough to the solution
- To help Newton's method in Position Analysis, you can take the starting point of the algorithm at time t_k to be the value of **q** from t_{k-1} (that is, the very previous configuration of the mechanism)
- See the pptx file available on the class website for MATLAB code that implements the Newton-Raphson method implemented in conjunction with the Position Analysis stage

End Kinematics

Start Dynamics

Purpose of Chapter 11

- At the end of this chapter you should understand what “dynamics” means and how you should go about carrying out a dynamics analysis

- We’ll learn a couple of things:
  - How to formulate the equations that govern the time evolution of a system of bodies in 3D motion
    - These equations are differential equations and they are called equations of motion
    - As many bodies as you wish, connected by any joints we’ve learned about…
  - How to compute the reaction forces in any joint connecting any bodies in the mechanism
  - Understand how to properly handle the applied, that is, external, forces to correctly use them in formulating the equations of motion

The Idea, in a Nutshell…

- **Kinematics**
  - You have as <u>many</u> constraints (kinematic and driving) as generalized coordinates
  - <u>No spare</u> degrees of freedom left
  - Position, velocity, acceleration found as the solution of algebraic problems (both nonlinear and linear)
  - We do not care whatsoever about forces applied to the system, we are told what the motions are and that's enough for the purpose of kinematics

- **Dynamics**
  - You only have a <u>few</u> constraints imposed on the system
  - You have <u>extra</u> degrees of freedom
  - The system evolves in time as a result of external forces applied on it
  - We very much care about forces applied and inertia properties of the components of the mechanism (mass, mass moment of inertia)

A Relevant Question…

- Dynamics **key** question: how can I get the acceleration of each body of the mechanism?
  - Why is acceleration so relevant? If you know the acceleration you can integrate it twice to get velocity and position information for each body
  - In other words, you want to get this quantity:

$$\ddot{\mathbf{q}}_i = \begin{bmatrix} \ddot{\mathbf{r}}_i \\ \ddot{\mathbf{p}}_i \end{bmatrix}$$

  - Alternatively, you can get first

$$\begin{bmatrix} \ddot{\mathbf{r}}_i \\ \dot{\bar{\omega}}_i \end{bmatrix}$$

    - Then use the fact that there is a relationship (haven't covered it yet)

$$\ddot{\mathbf{p}} \longleftrightarrow \dot{\bar{\omega}}$$

… and Its Answer

- The answer to the key question: To get the acceleration of each body, you first need to formulate the **equations of motion** (EOM)
  - Simplest possible form, ME240, motion of a particle: **F=ma**
    - Actually, the proper way to state this is **ma=F**, which is the “equation of motion” and is the most important piece of the dynamics puzzle
    - Back then, the acceleration would have been simply **a=F/m**

  - In ME751, first we’ll show that the EOM are obtained as

$$
\begin{aligned}
\mathbf{M}\ddot{\mathbf{r}} - \mathbf{F} &= \mathbf{0}_{3nb} \qquad \text{Equations of Motion governing translation} \\
\bar{\mathbf{J}}\dot{\bar{\omega}} - \bar{\mathbf{n}} + \tilde{\bar{\omega}}\bar{\mathbf{J}}\bar{\omega} &= \mathbf{0}_{3nb} \qquad \text{Equation of Motion governing rotation}
\end{aligned}
$$

  - This will be then revisited to formulate the equations of motion for a system of bodies interacting through contact, friction, and bilateral constraints, at the same time being subjected to the action of external forces