Quotes of the Day
[courtesy of Lijing]

山重水复疑无路，柳暗花明又一村。——陆游 《游山西村》
After endless mountains and rivers that leave doubt whether there is a path out, suddenly one encounters the shade of a willow, bright flowers and a lovely village.
--Lu You, 1125-1210 AD

不以物喜，不以己悲。——范仲淹《岳阳楼记》
Not pleased by external gains, not saddened by personal losses.
--Fan Zhongyan, 1046 AD

水至清则无鱼，人至察则无徒。——戴德《大戴礼记》
No fish can survive if the water is too clean, no companions will stay if the person is too astute.
--Dai De, 43 BC-33

兵者，诡道也。故能而示之不能，用而示之不用，近而示之远，远而示之近。——《孙子兵法》
All warfare is based on deception. Hence, when we are able to attack, we must seem unable; when using our forces, we must appear inactive; when we are near, we must make the enemy believe we are far away; when far away, we must make him believe we are near.
--Sun Wu, 《The Art of War》, 512 BC
Before we get started…

• Last time:
  • Wrapped up, use of BDF to solve 1DOF nonlinear IVP (warm up)
  • Started, numerical method for the solution of DAEs of multibody dynamics

• Today:
  • Loose ends, numerical method for the solution of DAEs of multibody dynamics
    • We’ll reference a cheat sheet, available online – summarizes all important identities we’re using in ME751
  • Start handling of frictional contact

• Looking ahead:
  • Comprehensive exam, coming up on November 21 (Monday) at 7 PM
    • Two hours long
    • Closed everything; you may bring one sheet of paper with any sort of information written on it (both sides)
    • Review session on Monday, November 21 during regular lecture hours (same room)
      • Compile a list of questions, I will try to answer them one by one
  • One page description of Final Project, using form that I’m going to make available on the class website
    • Due on Monday, October 24
  • Material covered during the rest of the semester
    • Handling of friction and contact (today and Friday)
    • The dynamics of flexible bodies, covered by Antonio Recuero (last nine lectures, starting Monday)
    • Last day of lecturing: November 11
Direct Approach: Step 2

[Slide from last time]

- It follows that if we discretize the combined EOM and level zero constraints we end up at time $t_n$ with the following equations:

\[
\begin{align*}
\mathbf{M} \ddot{\mathbf{r}}_n + \Phi^T_r(\mathbf{r}_n, \mathbf{p}_n, t_n)\lambda_n &= \mathbf{F}(\dot{\mathbf{r}}_n, \dot{\mathbf{p}}_n, \mathbf{r}_n, \mathbf{p}_n, t_n) \\
\mathbf{J}^p(\mathbf{p}_n) \ddot{\mathbf{p}}_n + \Phi^T_p(\mathbf{r}_n, \mathbf{p}_n)\lambda_n + \mathbf{P}^T(\mathbf{p}_n) \lambda^p_n &= \ mathbf{\hat{t}}(\dot{\mathbf{r}}_n, \dot{\mathbf{p}}_n, \mathbf{r}_n, \mathbf{p}_n, t_n) \\
\Phi^p(\mathbf{p}_n) &= 0_{nb} \\
\Phi(\mathbf{r}_n, \mathbf{p}_n, t_n) &= 0_{nc}
\end{align*}
\]

- IMPORTANT REMARK: Recall that above you see quantities depending on $\mathbf{r}_n$. Due to the BDF integration formulas, $\mathbf{r}_n$ is a function of $\dot{\mathbf{r}}_n$ (see STEP 1).

  - In other words, it’s like a “#define” statement in C or C++: wherever you see $\mathbf{r}_n$, substitute it by $\mathbf{r}_n = \mathbf{C}_n\dot{\mathbf{r}}_n(l) + \beta_0 h^2 \ddot{\mathbf{r}}_n$.

- When it comes to $\dot{\mathbf{r}}_n$, it should be replaced everywhere with the quantity $\mathbf{C}_n\dot{\mathbf{r}}_n(l) + \beta_0 h \dddot{\mathbf{r}}_n$.

- The same discussion applies when it comes to expressing $\mathbf{p}_n$ and $\dot{\mathbf{p}}_n$ in terms of $\ddot{\mathbf{p}}_n$. 
Direct Approach: Step 3
[Slide from last time]

- Recall how the discretized EOM for the mass-spring-damper system looked like in last lecture:

\[ g(a_n) = ma_n + cv_n^2 + kx_n^3 - \sin(2t_n) \]

- This time around, when dealing with a constrained multibody system, for the Ford F-150 solution methodology, the discretized constrained equations of motion look like this:

\[
\begin{bmatrix}
M\ddot{r}_n + \Phi_x^T(r_n, p_n, t_n)\lambda_n - F(\ddot{r}_n, \dot{p}_n, r_n, p_n, t_n) \\
J^p(\dot{p}_n)\ddot{p}_n + \Phi_p^T(r_n, p_n)\lambda_n + P^T(p_n)\lambda^p_n - \hat{f}(\ddot{r}_n, \dot{p}_n, r_n, p_n, t_n) \\
\frac{1}{\beta_k h^2} \Phi^p(p_n) \\
\frac{1}{\beta_k h^2} \Phi(r_n, p_n, t_n)
\end{bmatrix} = 0_{8n_b+nc}
\]

- Just like last time, there are several ways to solve this nonlinear system of equations based on the choice of the iteration matrix \( \Psi^{(\nu)} \)

  - Quasi-Newton: the simplest, this is going to be what you’ll use in your next HW
  - Newton Raphson: the iteration matrix is super messy due to several partial derivatives that are tricky to compute
  - Modified-Newton: basically Newton-Raphson, except that the iteration matrix is only evaluated and factored once at the beginning of the iterative process

- Irrespective of what flavor of Newton method you use, you’ll iterate in the following way:

\[
\begin{cases}
\Psi^{(\nu)} \cdot \Delta z^{(\nu)} = -g(z^{(\nu)}) \\
z^{(\nu+1)} = z^{(\nu)} + \Delta z^{(\nu)}
\end{cases}
\]

where I used the notation \( z = \begin{bmatrix} \ddot{r}_n \\ \ddot{p}_n \\ \lambda^p_n \\ \lambda_n \end{bmatrix} \)
The Newton-Raphson Iteration Matrix

[Step 3, Details]

\[
\begin{bmatrix}
\mathbf{M}\ddot{\mathbf{r}}_n + \mathbf{P}_r^T(r_n, \mathbf{p}_n, t_n)\lambda_n - \mathbf{F}(\dot{r}_n, \dot{p}_n, r_n, \mathbf{p}_n, t_n)
\end{bmatrix}
\begin{bmatrix}
\mathbf{J}_p^T(\mathbf{p}_n)\dot{\mathbf{p}}_n + \mathbf{P}_r^T(r_n, \mathbf{p}_n)\lambda_n + \mathbf{P}_p^T(p_n)\lambda_p - \mathbf{\tau}(\dot{r}_n, \dot{p}_n, r_n, \mathbf{p}_n, t_n)
\end{bmatrix}
\begin{bmatrix}
\frac{1}{\beta_0^2 h^2} \mathbf{\Phi}^p(\mathbf{p}_n)
\end{bmatrix}
\begin{bmatrix}
\frac{1}{\beta_0^2 h^2} \mathbf{\Phi}(r_n, \mathbf{p}_n, t_n)
\end{bmatrix}
= 0
\]

- The full blown Jacobian is as follows:

\[
\mathbf{\Psi} = \\
\begin{bmatrix}
\Psi_{11} & \Psi_{12} & 0_{3nb\times nb} & \mathbf{P}_r^T(r_n, \mathbf{p}_n, t_n) \\
\Psi_{21} & \Psi_{22} & \mathbf{P}^T & \mathbf{P}_p^T(r_n, \mathbf{p}_n, t_n) \\
0_{nb\times 3nb} & \mathbf{P} & 0_{nb\times nb} & 0_{nb\times nc} \\
\mathbf{\Phi}_r(r_n, \mathbf{p}_n, t_n) & \mathbf{\Phi}_p(r_n, \mathbf{p}_n, t_n) & 0_{nc\times nb} & 0_{nc\times nc}
\end{bmatrix}
\]

- The following notation has been used to fit the matrix above on one slide:

\[
\Psi_{11} \equiv \mathbf{M} + h^2 \beta_0^2 [\mathbf{P}_r^T(r_n, \mathbf{p}_n, t_n)\lambda_n]_r - h^2 \beta_0^2 \mathbf{F}_r(\dot{r}_n, \dot{\mathbf{p}}_n, r_n, \mathbf{p}_n, t_n) - h\beta_0 \mathbf{F}_r(\dot{r}_n, \dot{\mathbf{p}}_n, r_n, \mathbf{p}_n, t_n)
\]

\[
\Psi_{12} \equiv h^2 \beta_0^2 [\mathbf{P}_r^T(r_n, \mathbf{p}_n, t_n)\lambda_n]_p - h^2 \beta_0^2 \mathbf{F}_p(\dot{r}_n, \dot{\mathbf{p}}_n, r_n, \mathbf{p}_n, t_n) - h\beta_0 \mathbf{F}_p(\dot{r}_n, \dot{\mathbf{p}}_n, r_n, \mathbf{p}_n, t_n)
\]

\[
\Psi_{21} \equiv h^2 \beta_0^2 [\mathbf{P}_p^T(r_n, \mathbf{p}_n, t_n)\lambda_n]_r - h^2 \beta_0^2 \mathbf{\tau}(\dot{r}_n, \dot{\mathbf{p}}_n, r_n, \mathbf{p}_n, t_n) - h\beta_0 \dot{\mathbf{\tau}}(\dot{r}_n, \dot{\mathbf{p}}_n, r_n, \mathbf{p}_n, t_n)
\]

\[
\Psi_{22} \equiv \mathbf{J}_p^T(\mathbf{p}_n) + h^2 \beta_0^2 [\mathbf{J}_p^T(\mathbf{p}_n)\dot{\mathbf{p}}_n]_p + h^2 \beta_0^2 (\mathbf{P}^T\lambda_n)_p + h^2 \beta_0^2 [\mathbf{P}_p^T(r_n, \mathbf{p}_n, t_n)\lambda_n]_p - h^2 \beta_0^2 \dot{\mathbf{\tau}}_p(\dot{r}_n, \dot{\mathbf{p}}_n, r_n, \mathbf{p}_n, t_n) - h\beta_0 \dot{\mathbf{\tau}}_p(\dot{r}_n, \dot{\mathbf{p}}_n, r_n, \mathbf{p}_n, t_n)
\]
The Newton-Raphson and Modified-Newton Iteration Matrix

[Step 3, Details of the Details]

- Note that using Newton-Raphson or Modified-Newton to solve the nonlinear system $g(\dot{r}_n, \dot{p}_n, \lambda_n, \lambda_n^p) = 0_{8n_b+n_c}$ requires the evaluation of all the partial derivatives in $\Psi$. Specifically, we have to compute the partial derivatives that appear in $\Psi_{11}$, $\Psi_{12}$, $\Psi_{21}$, and $\Psi_{22}$.

- We need to provide the partial derivative $[J^p \dot{p}]_p = [4G^T \dot{J} G \dot{p}]_p$.

- Since we have only four basic GCONs, we can easily compute the following partial derivatives:

$$[\Phi_r^T (r_n, p_n, t_n) \lambda_n]_r \quad [\Phi_r^T (r_n, p_n, t_n) \lambda_n]_p \quad [\Phi_p^T (r_n, p_n, t_n) \lambda_n]_r \quad [\Phi_p^T (r_n, p_n, t_n) \lambda_n]_p$$

- When it comes to the forces and torques, we need to compute the following:

$$F_r(\dot{r}_n, \dot{p}_n, r_n, p_n, t_n) \quad \dot{F}_r(\dot{r}_n, \dot{p}_n, r_n, p_n, t_n) \quad \dot{F}_p(\dot{r}_n, \dot{p}_n, r_n, p_n, t_n) \quad \dot{F}_p(\dot{r}_n, \dot{p}_n, r_n, p_n, t_n)$$

$$\hat{F}_r(\dot{r}_n, \dot{p}_n, r_n, p_n, t_n) \quad \hat{F}_r(\dot{r}_n, \dot{p}_n, r_n, p_n, t_n) \quad \hat{F}_p(\dot{r}_n, \dot{p}_n, r_n, p_n, t_n) \quad \hat{F}_p(\dot{r}_n, \dot{p}_n, r_n, p_n, t_n)$$

- Since we don’t know in general the expression of the external forces/torques, computing the partial derivatives above is tricky, and it’s typically done by numerical differencing.

- This is for instance what happens in ADAMS for user defined forces and/or torques, when ADAMS does not have a priori knowledge about the expression of the forces and/or torques.

- We will not pursue this further in ME751.
Computing the Partial Derivative $[4G^T \bar{J}G\ddot{p}]_p$

- **IMPORTANT**: The discussion below concerns the computation of the partial derivative for one body $i$ of the mechanical system. Recall that one has to assemble the block diagonal matrix $[J^p \ddot{p}]_p$ from the derivative that we obtain on this slide.

  - In other words, you have to compute what we do in this slide $nb$ times, once for each body. Only at that point will you be able to assemble, one block at a time, the diagonal block matrix $[J^p \ddot{p}]_p$ that is associated with the *entire* mechanical system.

- First, note that for any arbitrary $b \in \mathbb{R}^4$,

  $$[G(p) \cdot b]_p = -G(b) \quad \Rightarrow \quad (\text{for } b = \ddot{p}) \quad \Rightarrow \quad [G(p) \cdot \ddot{p}]_p = -G(\ddot{p})$$

- Next, introduce the matrix $T$ that is obtained based on an arbitrary vector $a \in \mathbb{R}^3$ (see bottom of left column of the cheat sheet):

  $$T(a) = \begin{bmatrix} 0 & -a^T \\ a & -\tilde{a} \end{bmatrix}$$

- The noteworthy property of $T(a)$ is the following (it comes in relation to our old friend, matrix $G(p)$):

  $$G^T(p) \cdot a = T(a) \cdot p$$

- Finally, using the notation $a = \bar{J}G(p) \ddot{p}$ and applying the chain rule of differentiation, it follows that

  $$[J^p(p) \ddot{p}]_p = [4G^T(p)\bar{J}G(p)\ddot{p}]_p = -4G^T(p)\bar{J}G(\ddot{p}) + 4T(a) = 4(T(a) - G^T(p)\bar{J}G(\ddot{p}))$$
Computing the Partial Derivatives of the Reaction Forces

- In what follows we’ll focus on the following partial derivatives:

\[
[\Phi_r^T(r_n, p_n, t_n)\lambda_n]_r \quad [\Phi_r^T(r_n, p_n, t_n)\lambda_n]_p \quad [\Phi_p^T(r_n, p_n, t_n)\lambda_n]_r \quad [\Phi_p^T(r_n, p_n, t_n)\lambda_n]_p
\]

- Before diving in - two things will come in handy
  - First, recall that
    \[
d_{ij} = r_j + A_j s_j^Q - r_i - A_i s_i^{\prime P} = r_j + s_j^Q - r_i - s_i^{\prime P}
    \]
    - It follows that
    \[
    [d_{ij}]_{r_i, r_j, p_i, p_j} = [-I_3 \quad I_3 \quad -B(p_i, s_i^P) \quad B(p_j, s_j^Q)]
    \]
  - Second, a helper matrix \( K \) will be introduced (see cheat sheet provided a while ago). It is defined in conjunction with the partial derivative of the matrix \( B(p, \bar{a}) \), which itself is the partial derivative of \( A\bar{a} \)
    - Specifically, for an arbitrary vector \( b \in \mathbb{R}^3 \),
      \[
      K(\bar{a}, b) \equiv \frac{\partial [B^T(p, \bar{a}) \cdot b]}{\partial p} = 2 \begin{bmatrix} \bar{a}^T b & \bar{a}^T \bar{b} \\
      \bar{a} b & \bar{a} b^T + \bar{b} a^T - \bar{a}^T b \cdot I_{3 \times 3} \end{bmatrix}
      \]

The GCon-DP1 Case

- First, recall that
  \[ \Phi^{DP1}(i, a_i, j, a_j, f(t)) = a_i^T A_i^T A_j a_j - f(t) = a_i^T a_j - f(t) = 0 \]

- Next, recall that
  \[ \Phi^{DP1}_{r_i, r_j, p_i, p_j} = [0_{1 \times 3} \ 0_{1 \times 3} \ a_j^T \hat{B}(p_i, \bar{a}_i) \ a_i^T \hat{B}(p_j, \bar{a}_j)] \]

- Therefore, for a given scalar value \( \lambda \) (this will be Lagrange Multiplier, available to you)
  \[ (\Phi^{DP1}_{r_i, r_j, p_i, p_j})^T \lambda = \lambda \begin{bmatrix} 0_{3 \times 1} \\ 0_{3 \times 1} \\ \hat{B}^T(p_i, \bar{a}_i) a_j \\ \hat{B}^T(p_j, \bar{a}_j) a_i \end{bmatrix} \]

- It follows that the sensitivity of the reaction force with respect to the generalized coordinates is obtained as
  \[ [(\Phi^{DP1}_{r_i, r_j, p_i, p_j})^T \lambda]_{r_i, r_j, p_i, p_j} = \lambda \begin{bmatrix} 0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 4} & 0_{3 \times 4} \\ 0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 4} & 0_{3 \times 4} \\ 0_{4 \times 3} & 0_{4 \times 3} & \hat{K}(\bar{a}_i, a_j) & \hat{B}^T(p_i, \bar{a}_i) \hat{B}(p_j, \bar{a}_j) \\ 0_{4 \times 3} & 0_{4 \times 3} & \hat{B}^T(p_j, \bar{a}_j) \hat{B}(p_i, \bar{a}_i) & \hat{K}(\bar{a}_j, a_i) \end{bmatrix} \]
The GCon-DP2 Case

- First, recall that
  \[ \Phi^{DP2}(i, \tilde{a}_i, \tilde{s}_i^P, j, \tilde{s}_j^Q, f(t)) = a_i^T A_i^T d_{ij} - f(t) = a_i^T d_{ij} - f(t) = 0 \]

- Next, recall that
  \[ \Phi^{DP2}_{r_i, r_j, p_i, p_j}(a_i, d_{ij}) = a_i^T (d_{ij})_{r_i, r_j, p_i, p_j} + d_{ij}^T (a_i)_{r_i, r_j, p_i, p_j} \]
  \[ = \begin{bmatrix} -a_i^T & a_i^T & d_{ij}^T B(p_i, \tilde{a}_i) - a_i^T B(p_i, \tilde{s}_i^P) & a_i^T B(p_j, \tilde{s}_j^Q) \end{bmatrix} \]

- Therefore, for a given scalar value \( \lambda \) (this will be Lagrange Multiplier, available to you)
  \[
  (\Phi^{DP2}_{r_i, r_j, p_i, p_j})^T \lambda = \lambda \begin{bmatrix}
  -a_i \\
  a_i \\
  B^T(p_i, \tilde{a}_i) d_{ij} - B^T(p_i, \tilde{s}_i^P)a_i \\
  B^T(p_j, \tilde{s}_j^Q)a_i
  \end{bmatrix}
  \]

- It follows that the sensitivity of the reaction force with respect to the generalized coordinates is obtained as
  \[
  [(\Phi^{DP2}_{r_i, r_j, p_i, p_j})^T \lambda]_{r_i, r_j, p_i, p_j} = \lambda \begin{bmatrix}
  0_{3 \times 3} & 0_{3 \times 3} & -B(p_i, \tilde{a}_i) & 0_{3 \times 4} \\
  0_{3 \times 3} & 0_{3 \times 3} & B(p_i, \tilde{a}_i) & 0_{3 \times 4} \\
  -B^T(p_i, \tilde{a}_i) & B^T(p_i, \tilde{a}_i) & X & B^T(p_i, \tilde{a}_i)B(p_j, \tilde{s}_j^Q) \\
  0_{4 \times 3} & 0_{4 \times 3} & B^T(p_j, \tilde{s}_j^Q)B(p_i, \tilde{a}_i) & K(\tilde{s}_j^Q, a_i)
  \end{bmatrix}
  \]

- Notation used: matrix \( X \) above stands for
  \[
  X \equiv K(\tilde{a}_i, d_{ij}) - K(\tilde{s}_i^P, a_i) - B^T(p_i, \tilde{a}_i)B(p_i, \tilde{s}_i^P) - B^T(p_i, \tilde{s}_j^P)B(p_i, \tilde{a}_i)
  \]
The GCon-D Case

- First, recall that the GCon-D assumes the expression

\[ \Phi^D(i, s_i^P, j, s_j^Q, f(t)) = d_{ij}^T d_{ij} - f(t) = 0 \]

- Next, recall that

\[ \Phi^D_{r_i, r_j, p_i, p_j} = (d_{ij}^T d_{ij})_{r_i, r_j, p_i, p_j} = 2d_{ij}^T[d_{ij}]_{r_i, r_j, p_i, p_j} \]

\[ = \begin{bmatrix} -2d_{ij}^T & 2d_{ij}^T & -2d_{ij}^T B(p_i, s_i^P) & 2d_{ij}^T B(p_j, s_j^Q) \end{bmatrix} \]

- Therefore, for a given scalar value \(\lambda\) (this will be Lagrange Multiplier, available to you)

\[ (\Phi^D_{r_i, r_j, p_i, p_j})^T \lambda = 2\lambda \begin{bmatrix} -d_{ij} & d_{ij} \\ B^T(p_i, s_i^P) d_{ij} & -B(p_j, s_j^Q) d_{ij} \end{bmatrix} \]

- It follows that the sensitivity of the reaction force with respect to the generalized coordinates is obtained as

\[ [(\Phi^D_{r_i, r_j, p_i, p_j})^T \lambda]_{r_i, r_j, p_i, p_j} = 2\lambda \begin{bmatrix} \mathbf{I}_{3 \times 3} & -\mathbf{I}_{3 \times 3} & B(p_i, s_i^P) & -B(p_j, s_j^Q) \\ -\mathbf{I}_{3 \times 3} & \mathbf{I}_{3 \times 3} & -B(p_i, s_i^P) & B(p_j, s_j^Q) \\ B^T(p_i, s_i^P) & -B^T(p_i, s_i^P) & X & -B^T(p_i, s_i^P) B(p_j, s_j^Q) \\ -B(p_j, s_j^Q) & B(p_j, s_j^Q) & -B^T(p_j, s_j^Q) B(p_i, s_i^P) & Y \end{bmatrix} \]

- Notation used: matrix \(X\) above stands for

\[ X \equiv -K(s_i^P, d_{ij}) + B^T(p_i, s_i^P) B(p_i, s_i^P) \]

\[ Y \equiv K(s_j^Q, d_{ij}) + B^T(p_j, s_j^Q) B(p_j, s_j^Q) \]
The GCon-CD Case

• First, recall that the GCon-CD assumes the expression

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

• Next, recall that

$$\Phi^{CD}_{r_i, r_j, p_i, p_j} = (c^T d_{ij})_{r_i, r_j, p_i, p_j} = c^T[d_{ij}]_{r_i, r_j, p_i, p_j}$$

$$= \begin{bmatrix} -c^T & c^T & -c^T B(p_i, \bar{s}_i^p) & c^T B(p_j, \bar{s}_j^Q) \end{bmatrix}$$

• Therefore, for a given scalar value \( \lambda \) (this will be Lagrange Multiplier, available to you)

$$\Rightarrow (\Phi^{CD}_{r_i, r_j, p_i, p_j})^T \lambda = \lambda \begin{bmatrix} -c \\ c \\ -B^T(p_i, \bar{s}_i^p)c \\ B^T(p_j, \bar{s}_j^Q)c \end{bmatrix}$$

• It follows that the sensitivity of the reaction force with respect to the generalized coordinates is obtained as

$$[(\Phi^{CD}_{r_i, r_j, p_i, p_j})^T \lambda]_{r_i, r_j, p_i, p_j} = \lambda \begin{bmatrix} 0_{3\times3} & 0_{3\times3} & 0_{3\times4} & 0_{3\times4} \\ 0_{3\times3} & 0_{3\times3} & 0_{3\times4} & 0_{3\times4} \\ 0_{4\times3} & 0_{4\times3} & -K(\bar{s}_i^p, c) & 0_{4\times4} \\ 0_{4\times3} & 0_{4\times3} & 0_{4\times4} & K(\bar{s}_j^Q, c) \end{bmatrix}$$
Derived Newton-Euler equations of motion for a mechanical system

Solution must satisfy (a) kinematic, and (b) Euler parameterization constraints

Ended up with a set of DAEs. We solve them on a time grid

At each node of the time grid, we discretized the DAE using BDF
  - Discretization leads to nonlinear system
  - NR, MN, and QN can be used to solve this system of nonlinear equations

Next: discuss about prescribing initial conditions
  - You must start in a healthy (consistent) configuration for solution to make sense
The Dynamics Analysis Problem:
Prescribing Initial Conditions

• Make sure that you prescribe a set of initial positions and velocities $r_0, p_0, \dot{r}_0, \dot{p}_0$, that satisfy the level zero and one constraint equations (note that below $p_i,0$, for $i = 1, \ldots, nb$, represents the value of the Euler Parameter associated with body $i$ at time step $t_0$):

$$\Phi(r_0, p_0, t_0) = 0_{nc}$$

$$\Phi^P(p_0) = \begin{bmatrix} \frac{1}{2} p_{1,0}^T p_{1,0} - \frac{1}{2} \\ \cdots \\ \frac{1}{2} p_{nb,0}^T p_{nb,0} - \frac{1}{2} \end{bmatrix} = 0_{nb}$$

$$\Phi_r(r_0, p_0, t_0) \dot{r}_0 + \Phi_p(r_0, p_0, t_0) \dot{p}_0 = \dot{v}_0 \quad \& \quad P(p_0) \dot{p}_0 = 0_{nb}$$

• Compute $\ddot{r}_0$, $\ddot{p}_0$, $\lambda_0$, and $\lambda_0^P$ using the equations of motion combined with level two constraint equations

$$M \ddot{r}_0 + \Phi^T_r(r_0, p_0, t_0) \lambda_0 = F(\dot{r}_0, \dot{p}_0, r_0, p_0, t_0)$$

$$J^P(p_0) \ddot{p}_0 + \Phi^T_p(r_0, p_0, t_0) \lambda_0 + P^T(p_0) \lambda_0^P = \dot{\lambda}(\dot{r}_0, \dot{p}_0, r_0, p_0, t_0)$$

$$P(p_0) \ddot{p}_0 = \dot{\gamma}(p_0, p_0)$$

$$\Phi_r(r_0, p_0, t_0) \ddot{r} + \Phi_p(r_0, p_0, t_0) \ddot{p} = \dot{\gamma}(\dot{r}_0, \dot{p}_0, r_0, p_0, t_0)$$
In matrix-vector form (we’ve seen this before; color code: **BLUE** for known quantities, **RED** for unknown quantities) you’ll have to solve the following linear system (dropped the 0 subscripts to keep things simpler):

\[
\begin{bmatrix}
M_{3nb \times 4nb} & 0_{3nb \times nb} & \Phi_r^T \\
0_{nb \times 3nb} & J_P & P_T \\
0_{nb \times nb} & P & 0_{nb \times nc} \\
\Phi_r & \Phi_p & 0_{nc \times nb} \\
\end{bmatrix}
\begin{bmatrix}
\dot{\mathbf{r}} \\
\dot{\mathbf{p}} \\
\lambda_p \\
\lambda \\
\end{bmatrix}
= 
\begin{bmatrix}
\mathbf{F} \\
\hat{\mathbf{r}} \\
\gamma_p \\
\hat{\gamma} \\
\end{bmatrix}
\]

- If you don’t have correct values for \( \mathbf{r}_0, \mathbf{p}_0, \dot{\mathbf{r}}_0, \dot{\mathbf{p}}_0, \ddot{\mathbf{r}}_0, \ddot{\mathbf{p}}_0, \lambda_0, \) and \( \lambda_p^0 \) you will start off on the wrong foot: you won’t be able to get a correct solution.
  - The flow chart on the next slide starts with the assumption that you have a set of healthy initial conditions at \( t_0 \) at levels zero, one, and two.
Solving the Dynamics Problem: The Flow Chart

**STEP 0**: Prime new time step. Set: $\nu = 0$, $t_n = t_{n-1} + h$, $\mathbf{r}_n^{(0)} = \mathbf{r}_{n-1}$, $\mathbf{p}_n^{(0)} = \mathbf{p}_{n-1}$, $\lambda_n^{(0)} = \lambda_{n-1}$.

**STEP 1**: Compute position and velocity using BDF and most recent accelerations $\mathbf{\ddot{r}}_n^{(\nu)}$ and $\mathbf{\ddot{p}}_n^{(\nu)}$:

\[
\mathbf{r}_n^{(\nu)} = C_n^r(l) + \beta_0^2 h^2 \mathbf{\ddot{r}}_n^{(\nu)} \quad \mathbf{\dot{r}}_n^{(\nu)} = C_n^r(l) + \beta_0 h \mathbf{\ddot{r}}_n^{(\nu)} \\
\mathbf{p}_n^{(\nu)} = C_n^p(l) + \beta_0^2 h^2 \mathbf{\ddot{p}}_n^{(\nu)} \quad \mathbf{\dot{p}}_n^{(\nu)} = C_n^p(l) + \beta_0 h \mathbf{\ddot{p}}_n^{(\nu)}
\]

**STEP 2**: Compute the residual in the nonlinear system; i.e., evaluate $\mathbf{g}^{(\nu)}(\mathbf{r}_n^{(\nu)}, \mathbf{p}_n^{(\nu)}, \lambda_n^{(\nu)}, \lambda_n^{(\nu)}) = \mathbf{g}_n^{(\nu)}$.

**STEP 3**: Solve linear system $\Psi^{(\nu)} \Delta \mathbf{z}^{(\nu)} = -\mathbf{g}_n^{(\nu)}$ to get correction $\Delta \mathbf{z}_n^{(\nu)}$. You’ll use here an iteration matrix according to the Newton solver of your choice (NR, MN, or QN).

**STEP 4**: Improve the quality of the approximate solution: $\mathbf{z}^{(\nu+1)} = \mathbf{z}^{(\nu)} + \Delta \mathbf{z}^{(\nu)}$

**STEP 5**: Set $\nu = \nu + 1$. If the norm of the correction is small enough, go to **STEP 6**. Otherwise, go to **STEP 1** (unless you already took too many iterations and feel like this is not going anywhere in which case you bail out; most likely you either have (a) too large of a step size $h$, (b) your problem has discontinuities, or (c) you have a bug (very unlikely ;-))

**STEP 6**: Accept the accelerations and lambdas computed in **STEP 4** as your solutions. Using the accelerations, do yet one more time **STEP 1** to get level zero and one variables that are in sync with the accelerations. Save the value of the time step $t_n$ and the level zero, one, and two unknowns in an array since you want to plot the results at the end of simulation. At this point you just finished one integration step. Life is good. Go back to **STEP 0**.
End: The Dynamics Analysis Problem

Begin: Handling Friction and Contact in Multibody Dynamics
Putting Things in Perspective

- Up to this point, the concepts covered have been settled for a while
- Looking ahead, things are in flux – continue to be research topics
  - Handling of friction and contact
  - The dynamics of flexible bodies
- Material less structured
  - Reflects time constraints or limitations in my understanding
Mass $\times$ Acceleration = Force
Mass \times \text{Acceleration} = \text{Force}

- Coulomb friction coefficient - \( \mu \)

\[ m \ddot{v} = W + F + N + F_f \]

\[ |F_f| \leq \mu |N| \]

Reflect on this: friction force can assume a bunch of values (all of them smaller than \( \mu \times N \) though)
Additive Manufacturing (3D SLS Printing)

Courtesy of Professor Tim Osswald, Polymer Engineering Center, UW-Madison
3 Second Dynamics – 1 million spheres dropping in a bucket
[Commercial Software Simulation]
Selective Laser Sintering (SLS) Layering

<table>
<thead>
<tr>
<th>Granular Material</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$N$</td>
<td>1 186 185</td>
</tr>
<tr>
<td>$\rho$</td>
<td>930 [kg/m$^3$]</td>
</tr>
<tr>
<td>$\rho_{(\text{mean})}$</td>
<td>0.029 [mm]</td>
</tr>
<tr>
<td>$\rho_{(\sigma)}$</td>
<td>0.0075 [mm]</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Simulation</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>$\Delta t$</td>
<td>$5 \times 10^{-6}$ [s]</td>
</tr>
<tr>
<td>Sim. Time</td>
<td>49 Hours</td>
</tr>
</tbody>
</table>

7,800,000 contacts
46,800,000 unknowns
Using Simulation in 3D Printing of Clothes
Vehicle on granular terrain – GPU
Rigid-Fluid-Flexible Coupling: AKA Breakfast
Wave propagation in ordered granular material
Granular Material Ubiquitous

- More than 50% of bulk material in industry comes in granular form

- Granular dynamics relevant in many real life problems
  - additive manufacturing
  - nanoparticle self-assembly
  - pharmaceutical industry
  - composite materials
  - mining
  - formation of asteroids and planets
  - meteorite cratering
  - avalanche dynamics
  - etc.
Handling Frictional Contact

- “Frictional contact”: active area of research, challenging problem

- Topic relevant in applications in CAE/Video Gaming/Virtual Reality

- Quick remarks:
  - “Rigid body” model: simplified the dynamics problem significantly
  - Painlevé (late 1800’s) came up with simple examples that looked like paradoxes: there would be no solution for the time evolution of simple rigid body dynamics problems with contact and Coulomb friction
  - “Elastic body” model: addresses paradoxes but at heavy computational price
  - D. Steward: rigid body dynamics approach that resolved Painlevé’s paradox
Discrete Element Method: DEM-P & DEM-C

Computational Many-body Dynamics
Handling interactions between shapes

DEM-C
(complementarity)
(rigid-body contact)

DEM-P
(penalty)
(soft-body contact)

Optimization techniques
Collision detection
DEM: What’s That?

- DEM: Discrete Element Method
- Coined in conjunction with granular material simulations
  - Emphasizing it is not finite element method, the latter a continuum method
- DEM-P: using a penalty approach in simulating granular material
- DEM-C: using a complementarity approach for granular dynamics
  - Covered next lecture
- IMPORTANT: the “penalty” or “complementarity” approaches to handling friction not the purview of granular dynamics
  - Any of these two approaches can be used with any frictional contact problem, regardless of whether is associated with granular dynamics or not
General Comments, Penalty Approach

- In the “Penalty” universe, by and large, everything regarded as spheres or collections of spheres

- Approach is very commonly used in handling granular material
  - Called “Discrete Element Method”

- The DEM proceeds by using deformable body mechanics to understand what happens when two spheres are pressed against each other
  - Standard reference:

- This understanding is subsequently grafted to the general dynamics problem of rigid bodies flying in space and colliding with each other
  - When they collide, a fictitious spring-damper element is placed between the two bodies
    - Sometimes spring & damping coefficient based on continuum theory mentioned above
    - Sometimes values are guessed (calibration) based on experimental data
The Discrete Element Method (DEM)

- Depending on the normal relative velocity between bodies that experience a collision and their material properties, if there is no relative angular velocity, the collision is
  - Elastic, if the contact induced deformation is reversible and independent of displacement rate
  - Viscoelastic, if the contact induced deformation is irreversible, but the deformation is dependent on the displacement rate
  - Plastic, if collision leaves an involved particle permanently deformed but the deformation of a body is independent of the displacement rate
  - Viscoplastic, if impact is irreversible and similar to the viscoelastic contact but deformation depends on the displacement rate

- According to the dependency of the normal force on the overlap and the displacement rate, the force schemes can be subdivided into:
  - Continuous potential models (like Lennard-Jones, for instance)
  - Linear viscoelastic models (simple, used extensively, what we use here)
  - Non-linear viscoelastic models
  - Hysteretic models (see papers of L. Vu-Quoc, in ’DEM, Further Reading’ slide)
The “Penalty method” relies on a record (history) of tangential displacement $\delta_t$ to model static friction (see figure at right)
The “Penalty method” in Chrono: relies on a record (history) of tangential displacement to model static friction.

\[
F_n = f \left( \frac{\delta_n}{D_{\text{eff}}} \right) \left( k_n \delta_n n - \gamma m_{\text{eff}} v_n \right)
\]

\[
F_t = f \left( \frac{\delta_n}{D_{\text{eff}}} \right) \left( -k_t \delta_t - \gamma m_{\text{eff}} v_t \right)
\]

If \( |F_t| > \mu |F_n| \) then scale \( |\delta_t| \) so that \( |F_t| = \mu |F_n| \)

[Visualize this as creep.]
Direct Shear Analysis via Granular Dynamics
[using LAMMPS/LIGGGHTS and Chrono]

1800 uniform spheres
randomly packed

Particle Diameter:
D = 5 mm

Shear Speed:
1 mm/s

Inter-Particle Coulomb
Friction Coefficient:
μ = 0.5
(Quartz on Quartz)

Void Ratio
(dense packing):
e = 0.4

[J. Fleischmann]→
1800 uniform spheres randomly packed

Particle Diameter: $D = 5\, \text{mm}$

Shear Speed: $1\, \text{mm/s}$

Inter-Particle Coulomb Friction Coefficient: $\mu = 0.5$ (Quartz on Quartz)

Void Ratio (dense packing): $e = 0.4$