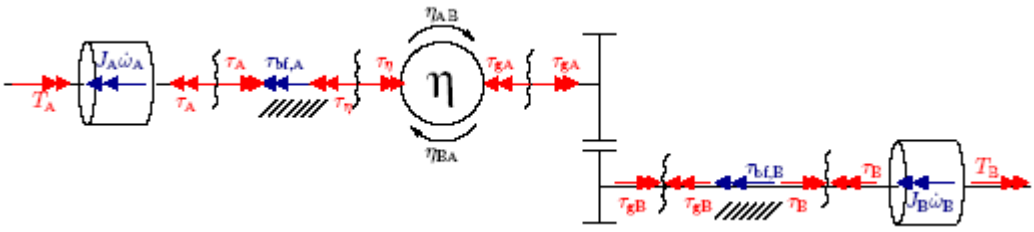
Problem with model „Lossy Gear“ and a proposed solution



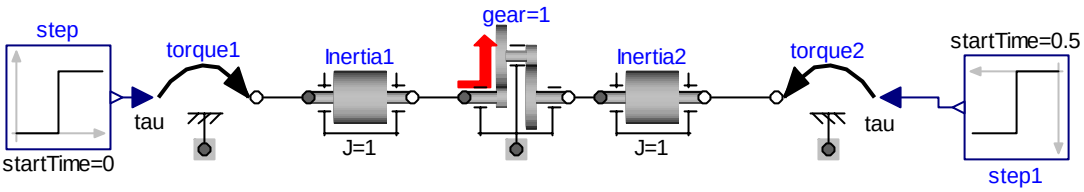
For the decision, which side of the gear is the driving one, the implementation within the MODELICA library uses the cut torque τ_A as described in the paper[1]:

$$\hat{\eta}_{mf} := \begin{cases} \eta_{mf1}(|\omega_A|) & : \begin{cases} \tau_A\,\omega_A > 0 \text{ or} \\ \tau_A = 0 \text{ and } \omega_A > 0 \end{cases} \\ 1/\eta_{mf2}(|\omega_A|) & : \begin{cases} \tau_A\,\omega_A < 0 \text{ or} \\ \tau_A = 0 \text{ and } \omega_A < 0 \end{cases} \\ \text{so that } \dot{\omega}_A = 0: & \omega_A = 0 \end{cases} \qquad (10)$$

Instead, the torque $\tau_\eta = \tau_A - \tau_{bf,A}$ should be used, that appears in equation (7):

$$\tau_B - \tau_{bf,B} = -i\,\hat{\eta}_{mf}\,(\tau_A - \tau_{bf,A})\,. \qquad (7)$$

The difference plays a role only in cases where the driving side is not obvious as following example demonstrates:



With the masses having positive angular velocities and the LossTable parameterized as $[0, 0.25, 0.25, 0.625, 2.5]$

This corresponds to $\eta_{mf1} = \eta_{mf2} = 0.5; \tau_{bfA} = \tau_{bfB} = 0.5$

The cut torque for $i = 1$ is to be:

$$\tau_A = \frac{J_B T_A - J_A T_B + J_A \hat{\tau}}{J_B - \hat{\eta} J_A} \quad (*)$$

A positive angular velocity leads to the decisions for what side is actually driving.

A is driving, if $\hat{\tau}_{bf,1} > \dfrac{-J_B T_A + J_A T_B}{J_A}$ and

B is driving, if $\hat{\tau}_{bf,2} < \dfrac{-J_B T_A + J_A T_B}{J_A}$.

[1] Pelchen C., Schweiger C., and Otter M.: "Modeling and Simulating the Efficiency of Gearboxes and of Planetary Gearboxes," in Proceedings of the 2nd International Modelica Conference, Oberpfaffenhofen, Germany, pp. 257-266, The Modelica Association and Institute of Robotics and Mechatronics, Deutsches Zentrum für Luft- und Raumfahrt e. V., March 18-19, 2002.

In the designed case, $T_A = 0, T_B = 1, J_A = J_B = 1$, both criteria are not fulfilled so that no side of the gear would be driving although the gear is in motion. This leads to a not converging solution in Dymola and following error occurs:

```
Expression gear.flange_a.tau >= 0 became true ( (gear.flange_a.tau)-
(0) = 0.3 )
  Variable gear_taux_0aPos = 1 at time 0.5
Shorter attempt for mixed equation system 0.002
Expression gear.flange_a.tau >= 0 became false ( (gear.flange_a.tau)-
(0) = -0.3 )
  Variable gear_taux_0aPos = 0 at time 0.5
Shorter attempt for mixed equation system 0.0020000000000000001
Fix point iteration did not converge at time : 0.5

ERROR: Finding consistent restart conditions failed at time:  0.5
```

Approach to solve the problem:

The cut torque $\tau_\eta = \tau_A \pm \tau_{bf,A}$ has to be used instead of τ_A in order to decide what side of the gear is the driving one.

The problem within the Modelica-code of „Lossy Gear“ is at the following lines of MODELICA-code:

```
  // Torque Losses
  tau_aPos = ideal or (flange_a.tau >= 0);
  tauLossMax = if tau_aPos then quadrant1 else quadrant2;
  tauLossMin = if tau_aPos then quadrant4 else quadrant3;
```

The Problem now is, that at the moment, at which the driving side has to be decided, the sign of ω (better: the direction of revolution decided by the variable `mode`) need not to be defined (for example when leaving stiction) and thus the sign of $\pm \tau_{bf,A}$ is not defined, which would be necessary to calculate $\tau_\eta = \tau_A \pm \tau_{bf,A}$.

In stuck-mode, both cases ($\omega > 0$ and $\omega < 0$) have to be regarded in order to decide what side would be driving in that case. Eventually, the direction of motion can be determined with the aid of the external torques.

The following calculation method for τ_η is now proposed by us: (Modification of the Lossy Gear model)

The main idea is:

- *Assume the direction of revolution (or stiction), by looking at the **two** cases (ω>0) and (ω<0)*
- *Determining the driving side for these two assumptions*
- *Now the real direction of revolution can be determined*
- *With this knowledge, the real driving side can be calculated*
- *Then the real friction due to bearing and mesh friction can be calculated.*
-

This is now translated in a modified Lossy gear model:

(the implementation is done in a way to change the original Lossy Gear model only slightly)
Reconstruct $\tau_{bf,A}$ (the absolute value of the bearing friction on the A side) from the friction and efficiency values given by the parameters:
tau_bf1=eta_mf1*tau_bf_a + 1/ratio tau_bf_B
tau_bf2=1/eta_mf2*tau_bf_a+ 1/ratio tau_bf_B

tau_bf_a=(tau_bf1-tau_bf2)/(eta_mf1-1.0/eta_mf2);
The analogous calculation has also to be performed for ω=0, to get the friction on the A-Side at stiction:
tau_bf_a_0:=(tau_bf1_0-tau_bf2_0)/(eta_mf1_0-1.0/eta_mf2_0);

Now tau_eta is calculated for the two separate cases (ω=0+) and (ω=0-)
Please note, that tau_bf_a_0 instead of tau_bf_a_0 has to be used, as these values are used only to decide the direction of revolution, which is only necessary for $\omega \approx 0$:

```
//assuming positive omega
tau_eta_p=flange_a.tau-tau_bf_a_0;
//assuming negative omegs
tau_eta_m=flange_a.tau+tau_bf_a_0;
```

Now The "quadrant-Calculations" have the be performed both for the limit cases (ω=0+) and (ω=0-):

```
// assuming w>=0, take value at w=0 to decide rolling/stuck mode
quadrant1_p = (1 - eta_mf1_0)*flange_a.tau + tau_bf1_0;
quadrant2_p = (1 - 1/eta_mf2_0)*flange_a.tau + tau_bf2_0;
tauLossMax_p = if noEvent(tau_eta_p>0) then quadrant1_p else quadrant2_p;

// assuming w<=0, take value at w=0 to decide rolling/stuck mode
quadrant4_m = (1 - 1/eta_mf2_0)*flange_a.tau - tau_bf2_0;
quadrant3_m = (1 - eta_mf1_0)*flange_a.tau - tau_bf1_0;
tauLossMin_m = if noEvent(tau_eta_m>0) then quadrant4_m else quadrant3_m;
```

and for the momentary angular velocity. (unchanged from the original Lossy gear)

```
quadrant1 = (1 - eta_mf1)*flange_a.tau + tau_bf1;
quadrant2 = (1 - 1/eta_mf2)*flange_a.tau + tau_bf2;
quadrant4 = (1 - 1/eta_mf2)*flange_a.tau - tau_bf2;
quadrant3 = (1 - eta_mf1)*flange_a.tau - tau_bf1;
```

Now the direction of revolution can be determined

```
// Determine rolling/stuck mode when w_rel = 0
startForward = pre(mode) == Stuck and sa > tauLossMax_p/unitTorque or initial()and w_a > 0
startBackward = pre(mode) == Stuck and sa < tauLossMin_m/unitTorque or initial()and w_a <0;
locked = not (ideal or pre(mode) == Forward or startForward or pre(mode)
       == Backward or startBackward);

   /* Finite state machine to fix configuration after the computation above
          The above equations are only dependent on pre(mode) and not on the actual
          value of mode. This prevents loops. So mode can be determined in one step. */
      mode = if ideal then Free else (if (pre(mode) == Forward or startForward)
          and w_a > 0 then Forward else if (pre(mode) == Backward or startBackward)
          and w_a < 0 then Backward else Stuck);
```

Now, with the direction of revolution known, tau_eta can be calculated:

```
//tau eta: only for determination of driving side for calculation of tauloss
tau_eta = if ideal
            then flange_a.tau
            else (if locked then flange_a.tau
                  else (if (startForward or pre(mode) == Forward)
                        then flange_a.tau-tau_bf_a
                        else  flange_a.tau+tau_bf_a));

    // Torque Losses
    tau_etaPos = ideal or (tau_eta >= 0);
```

The driving side decision must now be based on tau_eta>0 and not on tau_a>0

```
tauLossMax = if tau_etaPos then quadrant1 else quadrant
tauLossMin = if tau_etaPos then quadrant4 else quadrant3
```

The friction values used in quadrant1..4 however, are based on the momentary ω and not on ω=0.

Also tauloss has to be calculated the values based on the friction values for the momentary angular verlocity:

```
   tauLoss = if ideal then 0 else (if locked then sa*unitTorque else (if (startForward or
           pre(mode) == Forward) then tauLossMax else tauLossMin));
```

Open issues:

- Question: why not useModelica.Mechanics.Rotational.FrictionBase? Is the reason, that here we have |tau_bf_Stick|= |tau_bf(omega→0)|?

11.09.2009
Christian Bertsch Christian.Bertsch@de.bosch.com