Switching between static and dynamic frictions

`timeEvolutionEulersMethod`

x(t)

F(t)

`setContactForceToParticle();`

```
interaction[k].contact.addUpContactForceTorque();
sys->contact_force[p0] += f_contact_normal;
```

`computeVelocities`

Solve R***U***(t) + ***F***(t) = 0

Find velocities ***U***(t) = (**v**, ***ω***)

and shear rate dγ/dt

```
dimensionless_number = shear_rate_numerator/shearstress_hyd;
stokes_solver.solve(vel_hydro, ang_vel_hydro) etc.
```

`timeStepMove();`

Update positions

x(t+dt) = ***x***(t) + ***U***(t) dt

```
dt = disp_max/max_velocity;
displacement(i, velocity[i]*dt);
checkNewInteraction();
updateInteractions();
interaction[k].updateState(deactivated);
contact.incrementDisplacements();
interaction->calcRelativeVelocities();
```

sliding velocity: **V**s

Vs(t) = ***v***_i(t) - ***v***_j(t) + a_i ***n***(t)×***ω***_i(t) + a_j ***n***(t)×***ω***_j(t)

If dynamic friction, check Vs(t) is still the opposite direction to F_fric(t).
If not, switch to static friction: state = -2.

tangential spring for sliding static friction

ξ(t+dt) = ***ξ***(t) + **V**s(t) dt

```
incrementTangentialDisplacement();
```
[only static friction]

center-to-center distance *r*(t+dt)
normal vector ***n***(t+dt)

```
calcNormalVectorDistanceGap();
updateContactState(deactivated);
contact.calcContactInteraction();
```

If static friction, the friction force is computed here.

F_Nc(t+dt)

F_Tc(t+dt) = -k***ξ***(t+dt)

Apply friction law

```
f_contact_normal
f_contact_tan
(this->*frictionlaw)();
```

[The state of friction can be changed only in corrector]
- If static friction, check Ft < μs Fn is satisfied. If not, switch to dynamic friction: state = 3
- If soon after switching back to static friction (state = -2), compute disp_tan from F_fric = μ_d F_Nc and change to state = 2.

Dynamic friction force

If dynamic friction, the friction force is computed here.

t(t) = ***V***s(t)/|***V***s(t)|

t'(t) = ***t***(t) - ***t***(t)•***n***(t+dt) ***n***(t+dt)

F_Tc(t+dt) = -μ_d ***F***_Nc(t+dt) ***t***'(t)