Water Droplet Simulation

Math 164 Final Project

David Stachnik

◄ | ►

Motivation (Why Do People Care?)

Example Applications

- Internal Combustion Engines
- Fire Suppression/Spray Cooling
- Aerosols
- Acid Rain
- Cloud Physics
- Heavy Rain and Ice Accretion on Aircaft Wings
- Ink Jet Printing
- Applications in Medicine (e.g. inhaled particles, including asthma inhalers)
- Manufacturing — droplet by molten droplet method
- Microencapsulation

Examples from *Dynamics of Droplets* by Frohn and Roth (2000)

◄ | ►

Simplified Problem

From Prosperetti's 1977 paper

$$\frac{\partial U}{\partial t} + (U.\nabla)\,U = \frac{1}{\rho}\,\mathrm{div}\,\sigma$$

$$\nabla.U = 0$$

with

$$\sigma_{ij} = -p\,\delta_{ij} + \mu\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right)$$

U = velocity field
t = time
ρ = density
σ = stress tensor
p = pressure
μ = viscosity

Even More Simplified

Assume surface of the form

$$r(\theta, \phi) = R + a_n(t)\, Y_n^m(\theta, \phi)$$

to get a new ODE

$$a_n'' + 2\, b_{n0}\, a_n' + \omega_{n0}^2\, a_n + 2\, \beta_n\, b_{n0} \int_0^t Q_n(t - \tau)\, a_n'(\tau)\, d\tau = 0$$

R = average equilibrium radius

Y_n^m = spherical harmonic

a_n = amplitude

$b_{n0} = (n-1)(2n+1)\, \nu / R^2$

$\omega_{n0}^2 = n(n-1)(n+2)\, \zeta / \rho R^3$

$\beta_n = (n-1)(n+1)/(2n+1)$

Q_n is defined by its Laplace transform

$$\tilde{Q}_n(p) = \left[1 + \frac{1}{2}\, \mathcal{J}_{n+\frac{3}{2}}(q)\right]^{-1}$$

$q = R\, (p/\nu)^{1/2}$

$\mathcal{J}_{n+3/2}(q) = q\, I_{n+\frac{1}{2}}(q) / I_{n+\frac{3}{2}}(q)$

◄ | ►

Laplace Transformed Solution

- **Defined as:**

$$\text{LaplaceTransform}[f[t], t, p] = \int_0^\infty f(t)\, e^{-p\,t}\, dt$$

- **Dimensionless variables**

```
ν → ϵ        Viscosity
t → τ        Time
p → s        Laplace frequency

Initial Conditions
a_n (τ = 0) → 1
a_n ' (0) → u0
```

- **Yields**

```
atilde[s_, n_, ϵ_, u0_] := 1/s (1 + (u0 s - n (n - 1) (n + 2)) /
      (s^2 + 2 (n - 1) (2 n + 1) ϵ s + n (n - 1) (n + 2) + 2 (n - 1)^2 (n + 1) ϵ s Q[s, n, ϵ]));
Clear[Q]
Print["\n\n\n", "ã(s) = ",
  Style[1/s + 1/s ((u0 s - const)/(s^2 + ϵ s const + ϵ s Q[s, ϵ] const)) // TraditionalForm, FontSize → 18]]
Q[s_, n_, ϵ_] := Module[{q = Sqrt[s/ϵ]},
  (2 BesselI[n + 3/2, q]) / (2 BesselI[n + 3/2, q] - q BesselI[n + 1/2, q])]
```

$$\tilde{a}(s) = (s\, u0 - \text{const}) \,/\, \left(s \left(s^2 + \text{const}\, \epsilon\, s + \text{const}\, \epsilon\, Q(s, \epsilon)\, s\right)\right) + \frac{1}{s}$$

◂ | ▸

Numerical Inversion of Laplace Transform

- **Common Idea**

Exploit multiple-precision computing to eliminate round-off errors and compute solution by brute force.

- **Fixed Talbot Algorithm**

- **Main Idea**

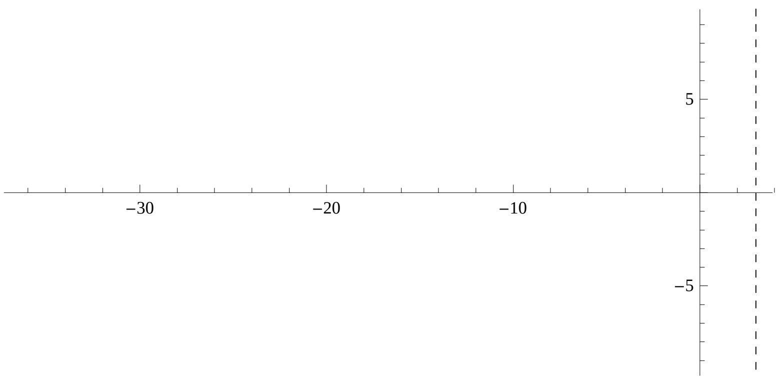
Deformation of the Bromwich integral:

$$\text{InverseLaplaceTransform}[F[p],\, p,\, t] = \frac{1}{2\pi i}\int_{\gamma - i\infty}^{\gamma + i\infty} F[p]\, e^{p\,t}\, dp$$

for some positive constant γ.

- **Deform the contour**

$$p(\theta) = \gamma\,\theta\,(\cot(\theta) + i) \qquad -\pi < \theta < +\pi$$



Then plug into inversion integral.
Approximate integral using trapezoidal rule with step size π/M

$$f(t, M) = \frac{\gamma}{M}\left(\frac{1}{2} F(\gamma)\exp(\gamma t) + \sum_{k=1}^{M-1} \text{Re}[\exp(t\, p(\theta_k))\, F(p(\theta_k))\, (p'(\theta_k)) / (i\,\gamma)]\right)$$

$$\theta_k = k\,\pi / M$$

By numerical experiment

$$\gamma = \frac{2\,M}{5\,t}$$

where M is the number of terms to be summed when calculating the integral.

- **The function**

```
FT[F_, t_, M_] := Module[{np, r, S, theta, sigma}, np = Max[M, $MachinePrecision];
  r = SetPrecision[2 M / (5 t), np];
  S = r theta (Cot[theta] + I);
  sigma = theta + (theta Cot[theta] - 1) Cot[theta];
  (r / M) Plus @@ Append[Table[Re[Exp[t S] (1 + I sigma) F[S]],
      {theta, Pi / M, (M - 1) Pi / M, Pi / M}], (1 / 2) Exp[r t] F[r]]]
```

Inversion Continued

- **Gaver Functionals (GWR) Algorithm**

Exact solution to inverse problem Post-Widder formula

$$\phi_k(t) = \frac{(-1)^k}{k!}\left(\frac{k}{t}\right)^{k+1} F^{(k)}\left(\frac{k}{t}\right)$$

where $F^{(k)}$ denotes the kth derivative and
$F(p) = \mathcal{L}[f(t)]$
Then $\phi_k(t) \to f(t)$ as $k \to \infty$

Approximate using difference formulas.

$$f_k(t) = \frac{\alpha k}{t}\binom{2k}{k}\sum_{j=0}^{k}(-1)^j\binom{k}{j}F((k+j)\alpha/t)$$

where $\alpha = \log(2)$.

Gaver functionals are a recursive method to calculate f_k but converge slowly.
Use Wynn rho convergence acceleration algorithm to get acronym GWR

■ The function

```
GWR[F_, t_, M_: 32, precin_: 0] := Module[
  {M1, G0, Gm, Gp, best, expr, τ = Log[2] / t, Fi, broken, prec},
  If[precin <= 0, prec = 21 M / 10, prec = precin];
  If[prec <= $MachinePrecision, prec = $MachinePrecision];
  broken = False;
  If[Precision[τ] < prec, τ = SetPrecision[τ, prec]];
  Do[Fi[i] = N[F[i τ], prec], {i, 1, 2 M}];
  M1 = M;
  Do[
   G0[n - 1] = τ (2 n) ! / (n! (n - 1) !) Sum[
      Binomial[n, i] (-1)^i Fi[n + i], {i, 0, n}];
   If[Not[NumberQ[G0[n - 1]]], M1 = n - 1; G0[n - 1] =.; Break[]];
   , {n, 1, M}];
  Do[Gm[n] = 0, {n, 0, M1}];
  best = G0[M1 - 1];

  Do[
   Do[
    expr = G0[n + 1] - G0[n];
    If[Or[Not[NumberQ[expr]], expr == 0], broken = True; Break[]];
    expr = Gm[n + 1] + (k + 1) / expr;
    Gp[n] = expr;
    If[OddQ[k],
     If[n == M1 - 2 - k, best = expr]
     ];
    , {n, M1 - 2 - k, 0, -1}];
   If[broken, Break[]];
   Do[Gm[n] = G0[n]; G0[n] = Gp[n], {n, 0, M1 - k}];
   , {k, 0, M1 - 2}];
  best
  ]
```

◄ | ►

The Results

```
Manipulate[Plot[GWR[atilde[#, 2 , ϵ, u0] &, t, ControlActive[10, 20]], {t, 0, 2},
  PlotPoints → ControlActive[10, 15], MaxRecursion → ControlActive[0, 2],
  PlotRange → {-.5, 1.1}, AxesLabel → {"τ", "a2(τ)"}],
 {{ϵ, 3 / 10}, 1 / 10, 1, 1 / 10}, {{u0, 3 / 10}, 0, 1, 1 / 10}]
```

◄ | ►

Pretty Pictures

- **Define functions**

```
a[0, ___] := 1;
a[t_, n_: 2, ϵ_: 3 / 10, u0_: 0, M_: 32] :=
 GWR[atilde[#, n, ϵ, u0] &, Rationalize[t, 10^-M], M]
Water = {Opacity[.8], Lighter[Blue, .35], Specularity[.7]};
r[t_, θ_, ϕ_, n_: 2, ϵ_: 3 / 10, u0_: 0, M_: 10] :=
  1 + a[Rationalize[t, 10^-M], n, ϵ, u0, M] SphericalHarmonicY[n, 0, θ, ϕ];
SetOptions[SphericalPlot3D, {PlotPoints → 15, MaxRecursion → 1, Mesh → None,
   Lighting → "Neutral", PlotStyle → Water, PlotRange → 2, Axes → False, Boxed → False}];
```

- **Spherical Harmonic Initial Conditions**

```
Manipulate[SphericalPlot3D[Evaluate[r[t, θ, ϕ, n, ϵ, u0, 10]],
  {θ, 0, π}, {ϕ, 0, 2 π}, Axes → False, Boxed → False], {{n, 2}, 2, 10, 1},
 {{ϵ, 3 / 10}, 1 / 10, 1, 1 / 10}, {{u0, 0}, 0, 1, 1 / 10}, {{t, 0}, 0, 2}]
```

- **Acorn-like Initial Condition**

```
acorn[t_, θ_, ϕ_, ϵ_: 3 / 10, u0_: 0, M_: 10] :=
 1 + Sum[(-1)^n .3/n^(5/4) a[Rationalize[t, 10^-M], 2 n + 1, ϵ, u0, M]
    SphericalHarmonicY[2 n + 1, 0, θ, ϕ], {n, 1, 13}]

acornlist = Table[SphericalPlot3D[
    Evaluate[acorn[t, θ, ϕ, 3 / 10, 0, 10]], {θ, 0, π}, {ϕ, 0, 2 π}], {t, 0, 3 / 2, 1 / 30}];

ListAnimate[acornlist]
```

◄ | ►

Still To Do

- **Verify results with Prosperetti paper**
- **Bubbles**
- **Examine numerical algorithms in more detail**
- **Try to understand where ODE comes from**