New Orbits for the Equimass n-Body Problem

Robert J. Vanderbei—The Optimization Guy

January 20, 2003
New Trends in Astrodynamics
Sponsors: NASA and Princeton Univ.

Operations Research and Financial Engineering, Princeton University

http://www.princeton.edu/∼rvdb

Optimization

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & b \leq h(x) \leq b + r, \\ & l \leq x \leq u \end{array}$$

- *Linear Programming (LP):* f and h are linear.
- *Convex Optimization:* f is convex, each h_i is concave, and $r = \infty$.
- *Nonlinear Optimization:* f and each h_i is assumed to be twice differentiable

- Generally, we seek a *local solution* in the vicinity of a given starting point.
- If problem is convex (which includes LP), any local solution is automatically a *global solution*.

Least Action Principle

Given: n bodies.

Let:
m_j denote the mass and
$z_j(t)$ denote the position in $\mathbb{R}^2 = \mathbb{C}$ of body j at time t.

Action Functional:

$$A = \int_0^{2\pi} \left(\sum_j \frac{m_j}{2} \|\dot{z}_j\|^2 + \sum_{j,k:k<j} \frac{m_j m_k}{\|z_j - z_k\|} \right) dt.$$

Equation of Motion

First Variation:

$$\delta A = \int_0^{2\pi} \sum_\alpha \left(\sum_j m_j \dot{z}_j^\alpha \dot{\delta z}_j^\alpha - \sum_{j,k:k<j} m_j m_k \frac{(z_j^\alpha - z_k^\alpha)(\delta z_j^\alpha - \delta z_k^\alpha)}{\|z_j - z_k\|^3} \right) dt$$

$$= -\int_0^{2\pi} \sum_j \sum_\alpha \left(m_j \ddot{z}_j^\alpha + \sum_{k:k\neq j} m_j m_k \frac{z_j^\alpha - z_k^\alpha}{\|z_j - z_k\|^3} \right) \delta z_j^\alpha dt$$

Setting first variation to zero, we get:

$$m_j \ddot{z}_j^\alpha = -\sum_{k:k\neq j} m_j m_k \frac{z_j^\alpha - z_k^\alpha}{\|z_j - z_k\|^3}, \qquad j = 1, 2, \ldots, n, \quad \alpha = 1, 2$$

Note: If $m_j = 0$ for some j, then the first order optimality condition reduces to $0 = 0$, which is *not* the equation of motion for a massless body.

Periodic Solutions

We assume solutions can be expressed in the form

$$z_j(t) = \sum_{k=-\infty}^{\infty} \gamma_k e^{ikt}, \qquad \gamma_k \in \mathbb{C}.$$

Writing with components $z_j(t) = (x_j(t), y_j(t))$ and $\gamma_k = (\alpha_k, \beta_k)$, we get

$$\begin{aligned} x(t) &= a_0 + \sum_{k=1}^{\infty} \left(a_k^c \cos(kt) + a_k^s \sin(kt) \right) \\ y(t) &= b_0 + \sum_{k=1}^{\infty} \left(b_k^c \cos(kt) + b_k^s \sin(kt) \right) \end{aligned}$$

where

$$\begin{array}{lll} a_0 = \alpha_0, & a_k^c = \alpha_k + \alpha_{-k}, & a_k^s = \beta_{-k} - \beta_k, \\ b_0 = \beta_0, & b_k^c = \beta_k + \beta_{-k}, & b_k^s = \alpha_k - \alpha_{-k}. \end{array}$$

The variables a_0, a_k^c, a_k^s, b_0, b_k^c, and b_k^s are the decision variables in the optimization model.

The AMPL Model

```
param N := 3;  # number of masses
param n := 15;  # number of terms in Fourier series representation
param m := 100;  # number of terms in numerical approx to integral

param theta {j in 0..m-1} := j*2*pi/m;

param a0 {i in 0..N-1} default 0;        param b0 {i in 0..N-1} default 0;
var as {i in 0..N-1, k in 1..n} := 0;    var bs {i in 0..N-1, k in 1..n} := 0;
var ac {i in 0..N-1, k in 1..n} := 0;    var bc {i in 0..N-1, k in 1..n} := 0;

var x {i in 0..N-1, j in 0..m-1}
  = a0[i]+sum {k in 1..n} ( as[i,k]*sin(k*theta[j]) + ac[i,k]*cos(k*theta[j]) );
var y {i in 0..N-1, j in 0..m-1}
  = b0[i]+sum {k in 1..n} ( bs[i,k]*sin(k*theta[j]) + bc[i,k]*cos(k*theta[j]) );

var xdot {i in 0..N-1, j in 0..m-1}
  = if (j<m-1) then (x[i,j+1]-x[i,j])*m/(2*pi) else (x[i,0]-x[i,m-1])*m/(2*pi);
var ydot {i in 0..N-1, j in 0..m-1}
  = if (j<m-1) then (y[i,j+1]-y[i,j])*m/(2*pi) else (y[i,0]-y[i,m-1])*m/(2*pi);

var K {j in 0..m-1} = 0.5*sum {i in 0..N-1} (xdot[i,j]^2 + ydot[i,j]^2);

var P {j in 0..m-1}
  = - sum {i in 0..N-1, ii in 0..N-1: ii>i}
          1/sqrt((x[i,j]-x[ii,j])^2 + (y[i,j]-y[ii,j])^2);

minimize A: (2*pi/m)*sum {j in 0..m-1} (K[j] - P[j]);
```

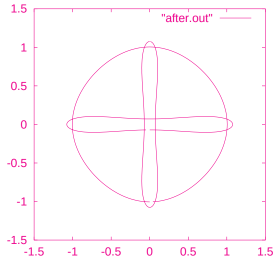
Continued...

```
let {i in 0..N-1, k in 1..n} as[i,k] := 1*(Uniform01()-0.5);
let {i in 0..N-1, k in 1..n} ac[i,k] := 1*(Uniform01()-0.5);
let {i in 0..N-1, k in n..n} bs[i,k] := 0.01*(Uniform01()-0.5);
let {i in 0..N-1, k in n..n} bc[i,k] := 0.01*(Uniform01()-0.5);

solve;
```

Choreographies and the Ducati

The previous AMPL model was used to find many *choreographies* (a la Moore and Montgomery/Chencinier) in the equimass n-body problem and the stable *Ducati* solution to the 3-body problem.



Limitations of the Model

- The infinite sum gets truncated to a finite sum. This amounts to adding constraints. Hence, the solution might be suboptimal. That is, the trajectory obtained might not satisfy the equations of motion.

- Masses must be positive.

- Model can't solve 2-body problem w/ eccentricity (see next slide).

Elliptic Solutions

An ellipse with semimajor axis a, semiminor axis b, and having its left focus at the origin of the coordinate system is given parametrically by:

$$x(t) = f + a\cos t, \qquad y(t) = b\sin t,$$

where $f = \sqrt{a^2 - b^2}$ is the distance from the focus to the center of the ellipse.
However, this is *not* the trajectory of a mass in the 2-body problem. Such a mass will travel faster around one focus than around the other. We need to introduce a time-change function $\theta(t)$:

$$x(t) = f + a\cos\theta(t), \qquad y(t) = b\sin\theta(t).$$

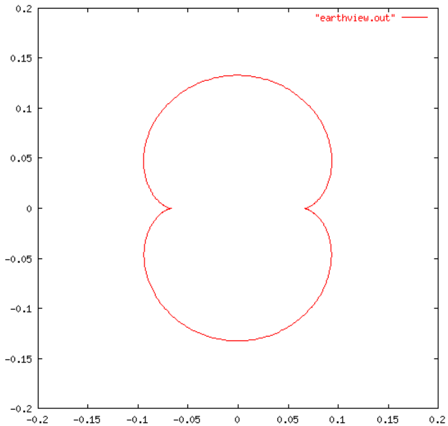
This function θ must be increasing and must satisfy $\theta(0) = 0$ and $\theta(2\pi) = 2\pi$.

The optimization model can be used to find (a discretization of) $\theta(t)$ automatically by changing `param theta` to `var theta` and adding appropriate monotonicity and boundary constraints.

A Hill-Type Solution...

...to the Eccentric Sun-Earth System

Using an eccentricity $e = f/a = 0.0167$ and appropriate Sun and Earth masses, we can find a periodic Hill-Type satellite trajectory in which the satellite orbits the Earth once per year.



Second Variation

$$
\begin{aligned}
\delta^2 A \; = \; & \int_0^{2\pi} \sum_j \sum_\alpha \left(\delta \dot{z}_j^\alpha \right)^2 dt \\
& + 3 \int_0^{2\pi} \sum_{j,k:j \neq k} \sum_{\alpha,\beta} \frac{(z_j^\alpha - z_k^\alpha)(z_j^\beta - z_k^\beta)(\delta z_j^\beta - \delta z_k^\beta)}{\|z_j - z_k\|^5} \delta z_j^\alpha dt \\
& - \int_0^{2\pi} \sum_{j,k:j \neq k} \sum_\alpha \frac{\delta z_j^\alpha - \delta z_k^\alpha}{\|z_j - z_k\|^3} \delta z_j^\alpha dt
\end{aligned}
$$

Sensitivity Analysis

Let

$$\xi^*(t) = \begin{bmatrix} x^*(t) \\ v^*(t) \end{bmatrix}$$

be a solution to

$$\dot{\xi} = A(\xi)$$

where

$$A\left(\begin{bmatrix} x(t) \\ v(t) \end{bmatrix}\right) = \begin{bmatrix} v(t) \\ a(x(t)) \end{bmatrix}$$

and

$$a(x) = \begin{bmatrix} a_1(x) \\ \vdots \\ a_n(x) \end{bmatrix}$$

and

$$a_j(x) = -\sum_{k:k\neq j} \frac{x_j - x_k}{\|x_j - x_k\|^2}, j = 1, 2, \ldots, n.$$

Sensitivity Analysis Continued

Consider a nearby solution $\xi(t)$:

$$\begin{aligned}\dot{\xi}(t) &= A(\xi(t)) \\ &\approx A(\xi^*(t)) + A'(\xi^*(t))(\xi(t) - \xi^*(t)) \\ &= \dot{\xi}^*(t) + A'(\xi^*(t))(\xi(t) - \xi^*(t)).\end{aligned}$$

Put $\Delta\xi = \xi - \xi^*$. Then $\dot{\Delta\xi} = A'(\xi^*(t))\Delta\xi$. A finite difference approximation yields

$$\begin{aligned}\Delta\xi(t+h) &= \Delta\xi(t) + hA'(\xi^*(t))\Delta\xi(t) \\ &= \left(I + hA'(\xi^*(t))\right)\Delta\xi(t).\end{aligned}$$

Iterating around one period, we get:

$$\Delta\xi(T) = \left(\prod_{i=0}^{n-1}\left(I + hA'(\xi^*(t_i))\right)\right)\Delta\xi(0),$$

where $h = T/n$ and $t_i = iT/n$.

Boring Perturbations—Invariants

Perturbations associated with invariants of the physical law are unimportant in calculating $\Delta\xi(T)$:

$$\begin{bmatrix} \Delta x \\ \Delta v \end{bmatrix} = \begin{bmatrix} e_1 \\ e_1 \\ e_1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} e_2 \\ e_2 \\ e_2 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ e_1 \\ e_1 \\ e_1 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ e_2 \\ e_2 \\ e_2 \end{bmatrix}, \begin{bmatrix} Rx_1 \\ Rx_2 \\ Rx_3 \\ Rv_1 \\ Rv_2 \\ Rv_3 \end{bmatrix}, \frac{1}{2}\begin{bmatrix} -3v_1 + 2x_1 \\ -3v_1 + 2x_1 \\ -3v_1 + 2x_1 \\ -3a_1 - v_1 \\ -3a_2 - v_2 \\ -3a_3 - v_3 \end{bmatrix},$$

where $R = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$.

These directions correspond to *translation*(2), *moving frame*(2), *rotation*, and *dilation*.

Dilation is explained on the next page.

All quantities are evaluated at $t = 0$.

Vector e_i denotes the i-th unit vector in $\mathbb{R}^2$.

Dilation

Consider spatial dilation by ρ together with a temporal dilation by θ:

$$y_j(t) = \rho x_j(t/\theta).$$

Given that the x_j's are a solution, it is easy to check that

$$\ddot{y}_j(t) = -\frac{\rho^3}{\theta^2} \sum_{k \neq j} \frac{y_j(t) - y_k(t)}{\|y_j(t) - y_k(t)\|^2}.$$

Hence, if mass is to remain fixed, we must have that $\rho^3 = \theta^2$:

$$y_j(t) = \rho x_j(t/\rho^{3/2}) \qquad \dot{y}_j(t) = \rho^{-1/2} v_j(t/\rho^{3/2}).$$

To find the perturbation direction corresponding to this dilation, we differentiate with respect to ρ at $\rho = 1$:

$$\frac{d}{d\rho} \left[\begin{array}{c} \rho x_j(t/\rho^{3/2}) \\ \rho^{-1/2} v_j(t/\rho^{3/2}) \end{array} \right] \Bigg|_{\rho=1} = \left[\begin{array}{c} -\frac{3}{2} v_j + x_j \\ -\frac{3}{2} a_j - \frac{1}{2} v_j \end{array} \right].$$

Projection

Put

$$P = \begin{bmatrix} e_1 & e_2 & 0 & 0 & Rx_1 & (-3v_1 + 2x_1)/2 \\ e_1 & e_2 & 0 & 0 & Rx_2 & (-3v_2 + 2x_2)/2 \\ e_1 & e_2 & 0 & 0 & Rx_3 & (-3v_3 + 2x_3)/2 \\ 0 & 0 & e_1 & e_2 & Rv_1 & (-3a_1 - v_1)/2 \\ 0 & 0 & e_1 & e_2 & Rv_2 & (-3a_2 - v_2)/2 \\ 0 & 0 & e_1 & e_2 & Rv_3 & (-3a_3 - v_3)/2 \end{bmatrix}.$$

For checking stability, we project any initial perturbation onto the null space of P^T using

$$\Pi = I - P(P^T P)^{-1} P^T.$$

From the fact that $x_1 + x_2 + x_3 = 0$ and $v_1 + v_2 + v_3 = 0$, it follows that all columns of P are mutually orthogonal *except* for the 5-th and 6-th columns. Hence, $P^T P$ is not a purely diagonal matrix.

Stability

We now focus on:

$$\Delta\xi(T) = \left(\prod_{i=0}^{n-1}\left(I + hA'(\xi^*(t_i))\right)\right)\left(I - P(P^TP)^{-1}P^T\right)\Delta\xi(0),$$

Stability: all eigenvalues of

$$\Lambda\Pi = \left(\prod_{i=0}^{n-1}\left(I + hA'(\xi^*(t_i))\right)\right)\left(I - P(P^TP)^{-1}P^T\right)$$

must be at most one in magnitude.

Numerical Results

Euler Stable Orbits

Name	$\max(\lambda_i(\Lambda))$	$\max(\lambda_i(\Lambda\Pi))$
Lagrange2	1.383	1.362
FigureEight3	1.228	4.220
Ducati3	1.105	3.885
Hill15	1.444	2.403
DoubleDouble5	12.298	12.298
DoubleDouble10	1.404	5.948
DoubleDouble20	1.890	1.890

Euler Unstable Orbits

Name	$\max(\lambda_i(\Lambda))$	$\max(\lambda_i(\Lambda\Pi))$
Lagrange3	81.630	81.630
OrthQuasiEllipse4	18.343	18.343
Rosette4	1.873	4.449
Braid4	727.508	711.811
Trefoil4	41228.515	41213.852
FigureEight4	221.642	194.095
FoldedTriLoop4	74758.355	74675.092
PlateSaucer4	3653.210	3653.210
BorderCollie4	188.235	188.052
Trefoil5	1.913e+8	1.917e+8
FigureEight5	2223.137	2223.457

Midpoint Integrator (using a Spring)

$$\ddot{x} = -x$$

Given: $x(0), v(0)$

Compute:

$$\begin{aligned} a(0) &= -x(0) \\ v(h/2) &= v(0) + (h/2)a(0) \end{aligned}$$

For $t = h, 2h, \ldots$

$$\begin{aligned} a(t) &= -x(t) \\ v(t+h/2) &= v(t-h/2) + ha(t) \\ x(t+h) &= x(t) + hv(t+h/2) \end{aligned}$$

t	x	v	a
0.0	1.000	0.000	-1.000
		-0.050	
0.1	0.995		-0.995
		-0.150	
0.2	0.980		-0.980
		-0.248	
0.3	0.955		-0.955
		-0.343	
0.4	0.921		-0.921
		-0.435	
0.5	0.877		-0.877

The Midpoint Integrator

```
if (integrator == MIDPOINT) {
  for (j=0; j<n; j++) {
      p[j].x += p[j].vx * dt;
      p[j].y += p[j].vy * dt;
  }
  for (j=0; j<n; j++) {
      p[j].ax = 0; p[j].ay = 0;
      for (i=0; i<n; i++) {
          if (i != j) {
              double r3 = dist3(p[i], p[j]);
              if (r3<r03) r3=r03;
              p[j].ax -= G * p[i].m * (p[j].x - p[i].x)/r3;
              p[j].ay -= G * p[i].m * (p[j].y - p[i].y)/r3;
          }
      }
  }
  for (j=0; j<n; j++) {
      p[j].vx += p[j].ax * dt;
      p[j].vy += p[j].ay * dt;
  }
}
```

Contents