AstDyn: Scientific Reference Manual

Algorithm Description and Mathematical Formulation

Michele Bigi
ITALOccult Project

December 10, 2025

Contents

Chapter 1

Introduction

This document serves as the scientific reference manual for the `AstDyn` library. It details the mathematical models and numerical algorithms implemented in the core engine. Unlike the API documentation (Doxygen), which focuses on software interfaces, this manual focuses on the Physics and Mathematics.

Chapter 2

Dynamical Model

2.1 Equations of Motion

The state vector $\mathbf{y} = [\mathbf{r}^T, \mathbf{v}^T]^T \in \mathbb{R}^6$ evolves according to:

$$\dot{\mathbf{y}} = \begin{bmatrix} \mathbf{v} \\ \mathbf{a}(\mathbf{r}, \mathbf{v}, t) \end{bmatrix} \tag{2.1}$$

The total acceleration $\mathbf{a}$ is the sum of:

- **Central Newtonian Gravity**: $\mathbf{a}_{2B} = -\frac{\mu}{r^3}\mathbf{r}$
- **N-Body Perturbations**: Sum of direct and indirect terms from planets.
- **Relativistic Corrections (1PN)**: Einstein-Infeld-Hoffmann equation approximation.
- **Solar Radiation Pressure**: Cannonball model.

Chapter 3

Numerical Integration Algorithms

3.1 Explicit Runge-Kutta-Fehlberg 7(8)

The RKF78 integrator is an explicit method with adaptive step size control. It uses 13 stages to produce a 7th order solution and an 8th order solution for error estimation.

Algorithm 1 RKF78 Integration Step

Require: Current state $t_n, \mathbf{y}_n$, Step size h
Ensure: Next state $t_{n+1}, \mathbf{y}_{n+1}$, Next step h_{new}

Compute Stages:
for $i = 1$ to 13 **do**
    $T_i = t_n + c_i h$
    $\mathbf{Y}_i = \mathbf{y}_n + h \sum_{j=1}^{i-1} a_{ij} \mathbf{k}_j$
    $\mathbf{k}_i = \mathbf{f}(T_i, \mathbf{Y}_i)$
end for
Update State: $\mathbf{y}_{n+1} = \mathbf{y}_n + h \sum_{i=1}^{13} b_i \mathbf{k}_i$
Error Estimation: $\hat{\mathbf{y}}_{n+1} = \mathbf{y}_n + h \sum_{i=1}^{13} \hat{b}_i \mathbf{k}_i$
$\epsilon = \|\mathbf{y}_{n+1} - \hat{\mathbf{y}}_{n+1}\|_\infty$
Step Size Control:
if $\epsilon \leq TOL$ **then**
    Accept step: $t_{n+1} = t_n + h$
    $h_{new} = h \cdot 0.9 \cdot \left(\frac{TOL}{\epsilon}\right)^{1/8}$
else
    Reject step: $t_{n+1} = t_n$
    $h_{new} = h \cdot 0.9 \cdot \left(\frac{TOL}{\epsilon}\right)^{1/8}$
    Repeat step with $h \leftarrow h_{new}$
end if

3.2 Implicit Gauss-Legendre Integrator

For symplectic integration, we solve the implicit Runge-Kutta equations using a simplified Newton-Raphson iteration.

Algorithm 2 Implicit Gauss-Legendre Step (Order $2s$)

Require: $t_n, \mathbf{y}_n, h$, Stages $s = 4$

Initialize stages $\mathbf{Z}_i^{(0)} = \mathbf{0}$ (or extrapolate from previous step)
Newton Iteration $k = 0 \dots k_{max}$:
for $i = 1$ to s **do**
  $\mathbf{Y}_i = \mathbf{y}_n + \mathbf{Z}_i^{(k)}$
  $\mathbf{R}_i = \mathbf{Z}_i^{(k)} - h \sum_{j=1}^{s} a_{ij} \mathbf{f}(t_n + c_j h, \mathbf{y}_n + \mathbf{Z}_j^{(k)})$
  Solve linear system (approx Jacobian $\mathbf{J} \approx \mathbf{I} - h\mathbf{A} \otimes \frac{\partial \mathbf{f}}{\partial \mathbf{y}}$) to find corrections $\Delta\mathbf{Z}$
  $\mathbf{Z}^{(k+1)} = \mathbf{Z}^{(k)} - \Delta\mathbf{Z}$
end for
if $\|\Delta\mathbf{Z}\| < TOL$ **then**
  Converged
  $\mathbf{y}_{n+1} = \mathbf{y}_n + h \sum_{i=1}^{s} b_i \mathbf{f}(t_n + c_i h, \mathbf{y}_n + \mathbf{Z}_i)$
  return
end if

Chapter 4

Orbit Determination Algorithms

4.1 State Transition Matrix (STM)

The STM $\mathbf{\Phi}(t, t_0) = \frac{\partial \mathbf{y}(t)}{\partial \mathbf{y}(t_0)}$ satisfies $\dot{\mathbf{\Phi}} = \mathbf{A}(t)\mathbf{\Phi}$. The calculation is performed simultaneously with the orbit integration.

Algorithm 3 Analytical STM Computation

Require: Current State $\mathbf{r}$
Ensure: Jacobian Matrix $\mathbf{G} = \frac{\partial \mathbf{a}}{\partial \mathbf{r}}$

Initialize $\mathbf{G} = \mathbf{0}_{3\times 3}$
Add 2-Body Term:
$r^{-5} = (\mathbf{r} \cdot \mathbf{r})^{-5/2}$
$\mathbf{G} \leftarrow \mathbf{G} + \frac{\mu}{r^5}(3\mathbf{r}\mathbf{r}^T - r^2\mathbf{I})$
Add N-Body Terms:
for each planet j **do**
  $\boldsymbol{\rho}_j = \mathbf{r} - \mathbf{r}_j$
  $\mathbf{G} \leftarrow \mathbf{G} + \mu_j \left(\frac{3\boldsymbol{\rho}_j \boldsymbol{\rho}_j^T}{\rho_j^5} - \frac{\mathbf{I}}{\rho_j^3} \right)$
end for

4.2 Differential Correction

The iterative least-squares procedure to correct the initial orbit.

Algorithm 4 Differential Correction Loop

Require: Initial Guess $\mathbf{y}_0$, Observations $\{\mathcal{O}_k\}$

$iter = 0$
while $iter < MAX\_ITER$ **do**
  Initialize Normal Matrix $\mathbf{B} = \mathbf{0}$, Gradient $\mathbf{D} = \mathbf{0}$
  Propagate state and STM to all observation epochs t_k
  for each observation k **do**
    Computed $\mathbf{C}_k = \mathrm{RaDec}(\mathbf{y}(t_k))$
    Residual $\boldsymbol{\xi}_k = \mathcal{O}_k - \mathbf{C}_k$
    Design Matrix $\mathbf{H}_k = \frac{\partial \mathbf{C}}{\partial \mathbf{y}} \boldsymbol{\Phi}(t_k, t_0)$
    Accumulate:
    $\mathbf{B} \leftarrow \mathbf{B} + \mathbf{H}_k^T W_k \mathbf{H}_k$
    $\mathbf{D} \leftarrow \mathbf{D} + \mathbf{H}_k^T W_k \boldsymbol{\xi}_k$
  end for
  Solve: $\delta \mathbf{y}_0 = \mathbf{B}^{-1} \mathbf{D}$
  Update: $\mathbf{y}_0 \leftarrow \mathbf{y}_0 + \delta \mathbf{y}_0$
  if $\mathrm{RMS}(\boldsymbol{\xi})$ converged OR $\|\delta \mathbf{y}_0\| < \epsilon$ **then**
    return $\mathbf{y}_0, \mathbf{P}_{cov} = \mathbf{B}^{-1}$
  end if
  $iter \leftarrow iter + 1$
end while