Homework 3 - Monte Carlo Analysis

Jose Eduardo Laruta Espejo
Facultad de Ingeniería - Universidad Mayor de San Andrés

June 3, 2019

1 Projectile model

We are going to use the following model for our projectile:

$$\sum F = Ma \tag{1}$$

If we take the x and y components and the gravity effect into account we have:

$$Ma_x = -F_x \tag{2}$$
$$Ma_y = -F_y - F_g \tag{3}$$
$$F_x = F\cos\theta_0 \tag{4}$$
$$F_y = F\sin\theta_0 \tag{5}$$
$$F = Bv + D|v|^2 \tag{6}$$

our states are given by:

$$x_1 = y \qquad x_2 = v_y \qquad x_3 = x \qquad x_4 = v_x$$

then, our state space form model is as follows:

$$\dot{x_1} = x_2 \tag{7}$$
$$\dot{x_2} = -\frac{(Bv + D|v|^2)cos(\theta)}{M} - g \tag{8}$$
$$\dot{x_3} = x_4 \tag{9}$$
$$\dot{x_4} = -\frac{(Bv + D|v|^2)sin(\theta)}{M} \tag{10}$$

where $v = \sqrt{x_2^2 + x_4^2}$ the current magnitude of the velocity and θ the angle between the horizontal axis and the velocity vector. The initial conditions are defined statistically by

$E[\theta_0] = 45[deg]$, $E[v_0] = 200[m/s]$ and standard deviation of 3%. Then, the initial condition vector given θ_0 and v_0 are:

$$x_0 = \begin{bmatrix} 0 \\ v_0 \sin\theta_0 \\ 0 \\ v_0 \cos\theta_0 \end{bmatrix}$$

the model expressed in matlab code:

```
function xdot = projectile_model(t,x)
    % Model constant and parameters
    global theta0;
    B = 0.01;         %N sec/m
    D = 3.0e-5;       % N sec^2/m^2
    M = 1;            % Kg
    g = 9.8;          %m/s^2
    % Compute useful values
    v = sqrt((x(2)^2) + (x(4)^2));
    theta = atan2(x(2), x(4));
    F = B*v + D*abs(v)^2;
    Fx = F * cos(theta);
    Fy = F * sin(theta);
    % State space
    xdot(1) = x(2);
    xdot(2) = -Fy - g;
    xdot(3) = x(4);
    xdot(4) = -Fx;
    % reset velocity if hit the ground
    if (x(1) <=0) && (t > 0)
        xdot(1) = 0;
        xdot(3) = 0;
    end
    xdot = xdot(:);
end
```

2 Monte Carlo Simulation

The code for the Simulation is as follows:

```

tspan = [0 40]; %reasonable time for steady state
% random initial condition
m_v0 = 200;              % mean for initial velocity
sigma_v0 = 0.03;         % sigma for initial velocity
m_theta0 = deg2rad(45); % mean for initial angle (radians)
sigma_theta0 = 0.03;     % sigma for initial angle
lambda = 3;              % kurtosis (3 = gaussian)
n_sig = 1.96;            % n_sigma
% for montecarlo simulation
n_trials = 10000;         % runs
stats_idx = 1;           % index for fill in the statistics
```

```
final_d = []              % empty array
hold on;
title('ballistic trajectory problem')
xlabel('range[m]')
ylabel('height[m]')
t_1 = [0 3300]; x_1 = [0 0];
plot(t_1,x_1,'-')
for q = 1:n_trials
    % simulation parameters
    v0 = m_v0 + sigma_v0*randn;      % random initial velocity
    theta0 = m_theta0 + sigma_theta0*randn; % random initial angle
    % initial states for simulation;
    x0 = [0; v0*sin(theta0); 0; v0*cos(theta0)];    % initial states
    [t, x] = ode45('projectile_model', tspan, x0);  % simulate
    final_d = [final_d; x(length(t), 3)];           % save final distance

    if mod(q, 100) == 0 % every 100 trials
        % plot trajectory
        plot(x(:,3), x(:,1));
        % compute statistics
        m_hat(stats_idx) = sum(final_d) / q
        dif_sq = (final_d - m_hat(stats_idx)).^2;
        p_hat = sum(dif_sq) / (q -1);
        debias = q/(q-1);
        sigma(stats_idx) = sqrt(debias*p_hat);          % save sigma
        % confidence limits for sigma
        sigma_low(stats_idx) = sigma(stats_idx)/(1 + n_sig * sqrt((lambda
            -1)/q));
        sigma_high(stats_idx) = sigma(stats_idx)/(1 - n_sig * sqrt((lambda
             -1)/q));
        % confidence limits for m
        m_low(stats_idx) = m_hat(stats_idx) - (n_sig * sigma(stats_idx))/
            sqrt(q);
        m_high(stats_idx) = m_hat(stats_idx) + (n_sig * sigma(stats_idx))/
            sqrt(q);
        stats_idx = stats_idx + 1;
    end
end
hold off
%%
% plot statistics
% mean distance
figure
t_plot = 1:length(sigma);

plot(t_plot, m_hat, t_plot, m_high, '+', t_plot, m_low, '+')
title('mean impact distance')
xlabel('cummulative set number x100')
ylabel('mean impact distance')
% sigma
figure

plot(t_plot, sigma, t_plot, sigma_high, '+', t_plot, sigma_low, '+')
title('sigma impact distance')
```

```
xlabel('cummulative set number x100')
ylabel('sigma impact distance')
```

after running the simulation the resulting plots are:

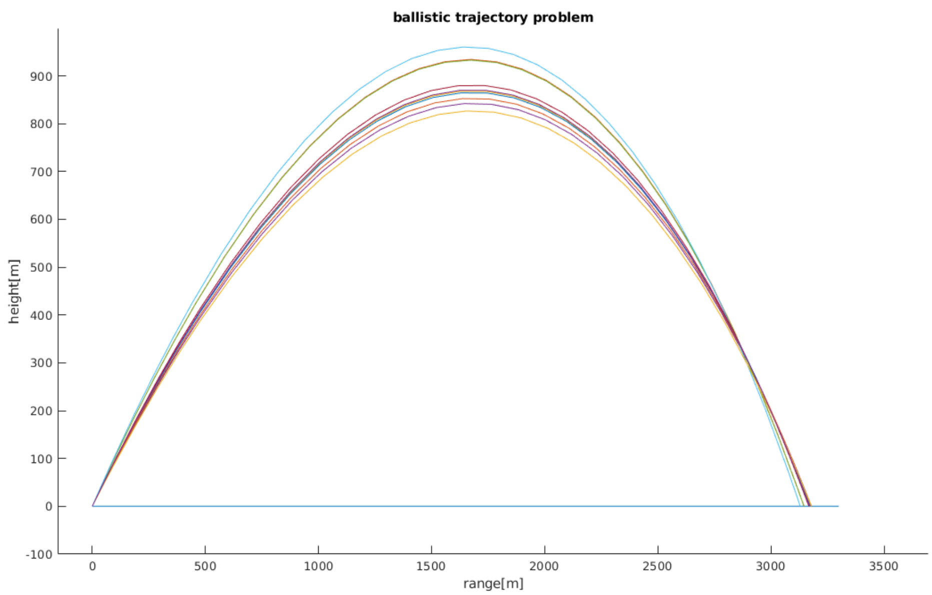

Figure 1: Simulation of the model

In the case of the simulation of the projectile trajectories, Fig 1 shows clearly the variation due to random initial conditions.

For the statistics we can se clearly the convergence of the mean as soon as the Monte Carlo trials begin to increase, the same for the standard deviation. In Fig 2 the mean is slowly converging and the interval of confidence decrease in size. The same goes for Sigma in Fig 3. Just for fun, the plots for 10000 iteration are shown in Figs 4 and 5, were we can see the convergence of the mean in 3159 m and sigma in 19.62 m.

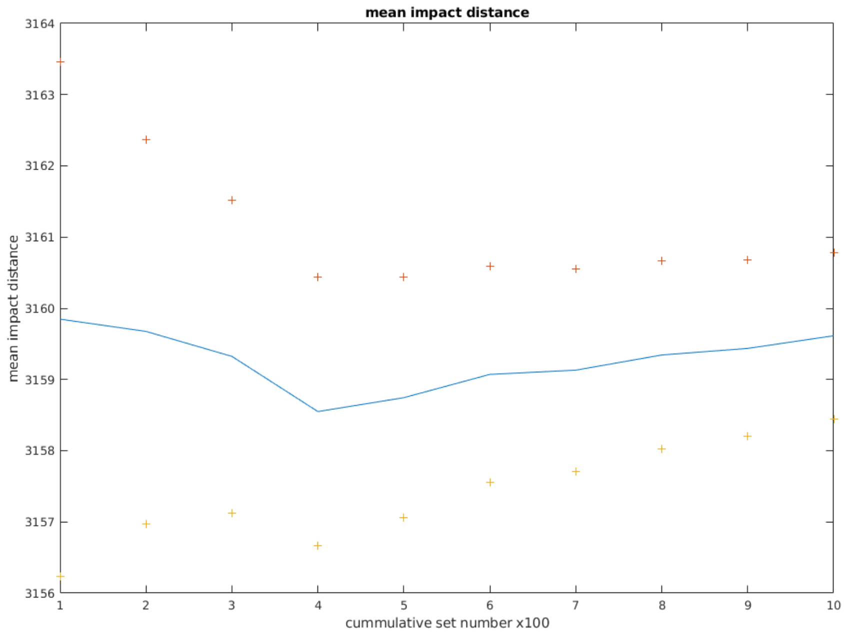

Figure 2: Mean convergence 1000 runs

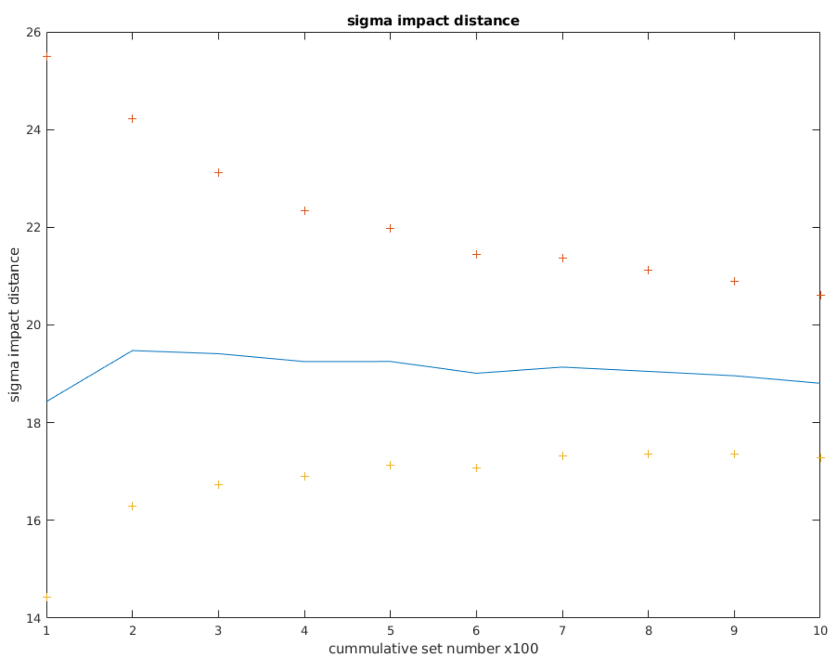

Figure 3: Sigma convergence 1000 runs

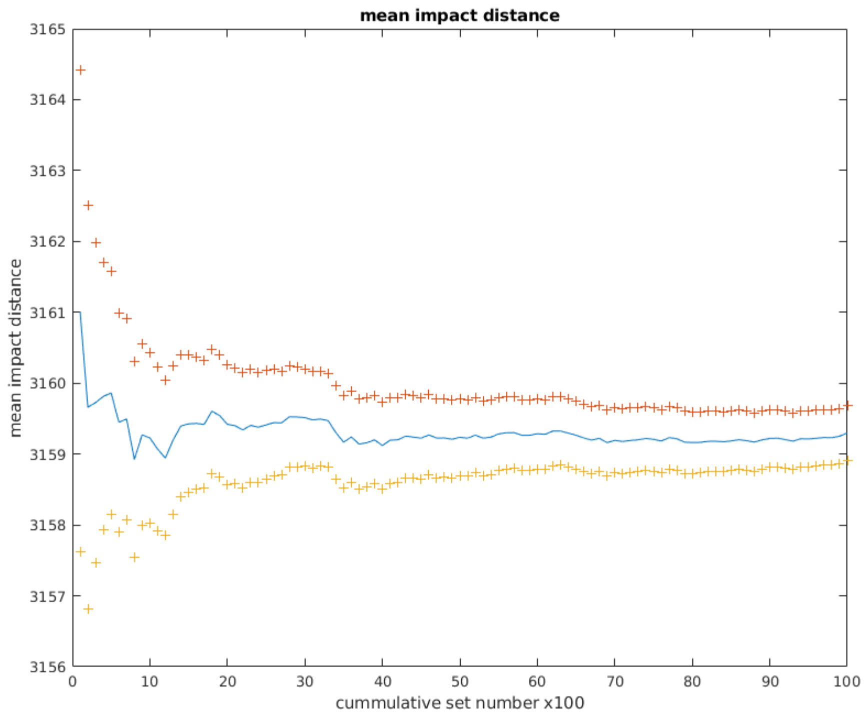

Figure 4: Mean convergence 10000 runs

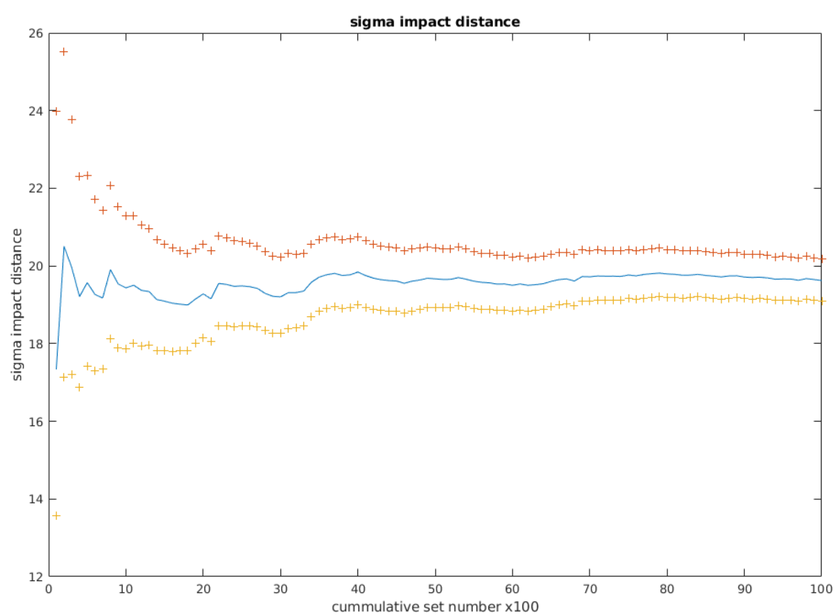

Figure 5: Sigma convergence 10000 runs