ASTR513 Homework2

Joshua Lothringer, Yifan Zhou

March 9, 2016

1. From measurements to inferences

In class, we studied the problem of projectile motion in a vertical, uniform gravitational field $\vec{g}$ for which we want to infer the two components of the initial velocity vector of the projectile, u_x and u_y given a measurement of the height h and range R of the motion. Repeat this investigation in this homework problem with the additional complication that our knowledge of the magnitude of the gravitational field is not perfect.
In particular, assume that the uncertainties in our measurements of the height and range of the projectile are well approximated by Gaussian functions with means and standard deviations of ($h_0 = 1.0\,\mathrm{m}, \sigma_h = 0.2\,\mathrm{m}$) and ($R_0 = 10.0\,\mathrm{m}, \sigma_R = 0.2\,\mathrm{m}$), respectively. Also assume that the uncertainty in our knowledge of the magnitude of the gravitational field is also well described by a Gaussian with a mean and standard deviation of ($g_0 = 9.81\,\mathrm{ms}^{-2}, \sigma_g = 0.05\,\mathrm{ms}^{-2}$).

(a). follow the traditional approach to error propagation to infer the uncertainties in the inference of u_x and u_y, given the measurement uncertainties.

$$u_x = \sqrt{\frac{g}{2h}}\frac{R}{2} \tag{1}$$

$$u_y = \sqrt{2gh} \tag{2}$$

Using error propagation

$$\sigma_{u_x}^2 = \frac{g}{8h}\sigma_R^2 + \frac{R^2}{32gh}\sigma_g^2 + \frac{gR^2}{32h^3}\sigma_h^2 \tag{3}$$

$$\sigma_{u_y}^2 = \frac{h}{2g}\sigma_g^2 + \frac{g}{2h}\sigma_h^2 \tag{4}$$

Thus

$$\sigma_{u_x} = 1.13\,\mathrm{ms}^{-1} \tag{5}$$

$$\sigma_{u_y} = 0.43\,\mathrm{ms}^{-1} \tag{6}$$

(b). Following frequentist arguments, use Monte Carlo realizations of the three measurements (h,R,and g) based on the distribution of their uncertainties to investigate the correlation in the uncertainties of the inferred quantities (u_x and u_y). Use the same Monte Carlo realizations to generate the one-dimensional, marginalized distributions over ux and uy and compare them to the results of part (a).

Monte Carlo result is presented in Figure 1. u_x and u_y are tightly correlated. From the marginal distribution of u_x and u_y, u_x is skewed to large velocity, and u_y is skewed to small velocity.

(c). Repeat part (b) analytically using (Jacobian) transformations of the probability distributions of the measurement. Compare your results to the Monte Carlo simulations.

$$P(u_x,u_y,g)\mathrm{d}u_x\mathrm{d}u_y\mathrm{d}g = G(h,\sigma_h)G(R,\sigma_R)G(g,\sigma_g)\mathrm{d}h\mathrm{d}R\mathrm{d}g \tag{7}$$

$$P(u_x,u_y,g) = G(h,\sigma_h)G(R,\sigma_R)G(g,\sigma_g)\left|J\left(\frac{h,R,g}{u_x,u_y,g}\right)\right| \tag{8}$$

Calculate the Jacobian determinant $J\left(\frac{h,R,g}{u_x,u_y,g}\right)$

$$\begin{aligned} J\left(\frac{h,R,g}{u_x,u_y,g}\right) &= \begin{vmatrix} \frac{\partial h}{\partial u_x} & \frac{\partial h}{\partial u_y} & \frac{\partial h}{\partial g} \\ \frac{\partial R}{\partial u_x} & \frac{\partial R}{\partial u_y} & \frac{\partial R}{\partial g} \\ \frac{\partial g}{\partial u_x} & \frac{\partial g}{\partial u_y} & \frac{\partial g}{\partial g} \end{vmatrix} \\ &= -\frac{2u_y^2}{g^2} \end{aligned} \tag{9}$$

$$P(u_x,u_y,g) = G(h,\sigma_h)G(R,\sigma_R)G(g,\sigma_g)\frac{2u_y^2}{g^2} \tag{10}$$

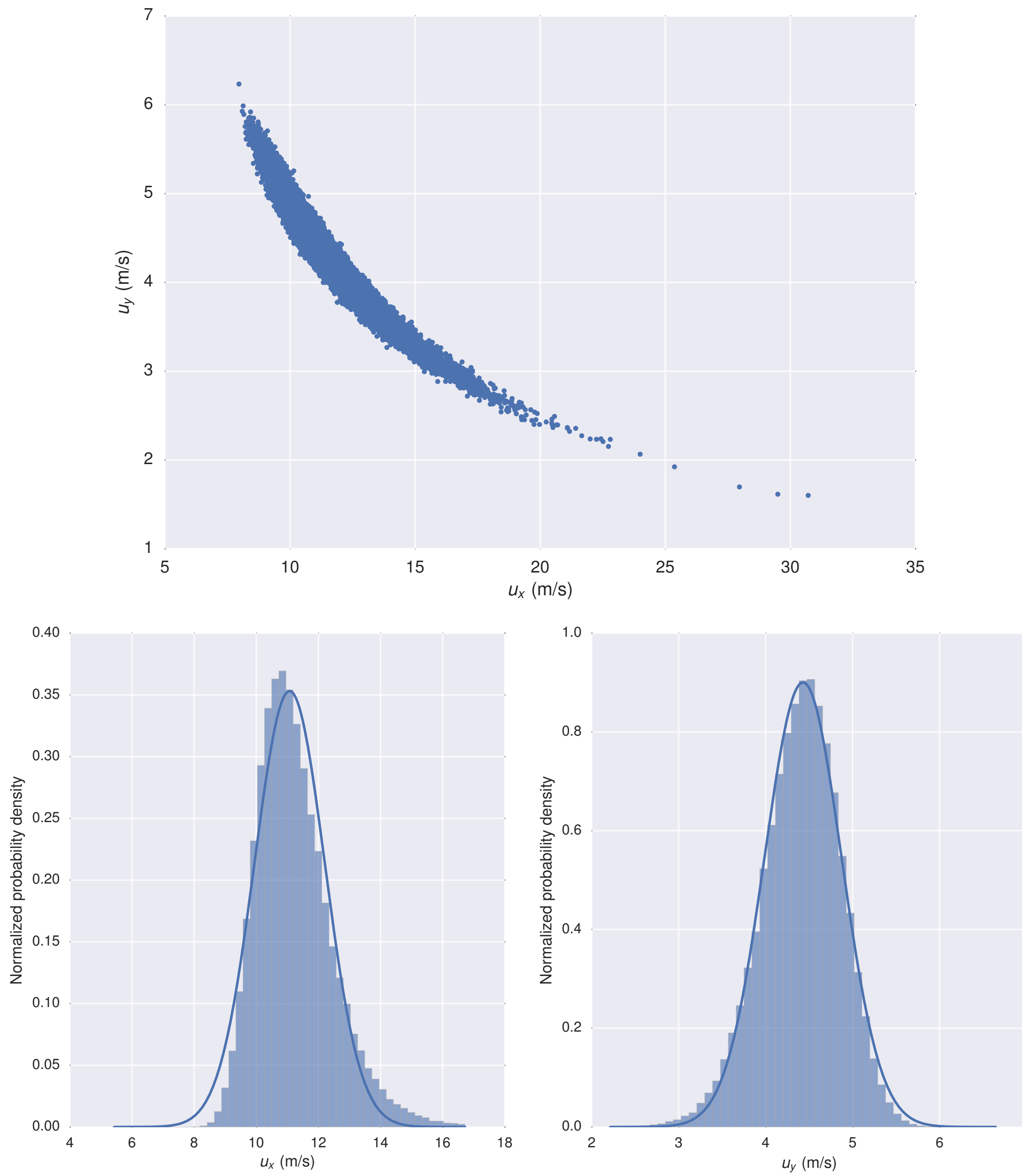

Figure 1: Joint distribution and marginal distributions of u_x, and u_y.

Using numerical Integration to calculate $P(u_x)$ and $P(u_y)$:

$$P(u_x) = \int \mathrm{d}u_y \int \mathrm{d}g P(u_x, u_y, g) \tag{11}$$

$$P(u_y) = \int \mathrm{d}u_x \int \mathrm{d}g P(u_x, u_y, g) \tag{12}$$

Plot the two semi-analitical result with Monte Carlo realization (Figure 2). Analytical calculation gives identical results to MC realization.

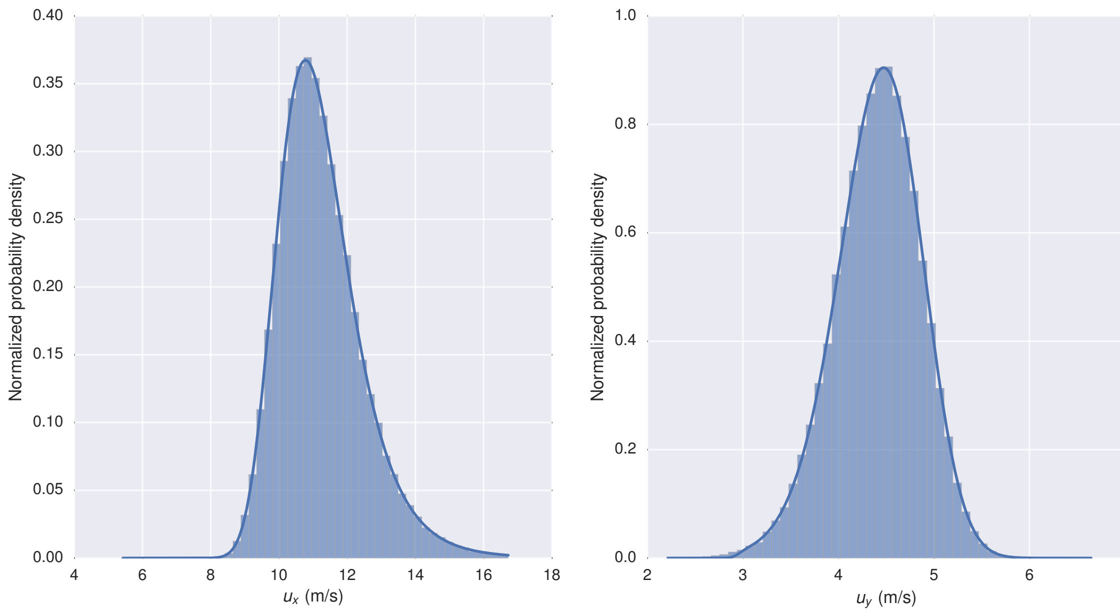

Figure 2: Compare MC realization and analytical calculation using Jacobian transformation

(d). Solve the same problem using Bayesian arguments and compare the results to that of the frequentist approach. As a first attempt, use priors over ux and uy that are constant.

(e). Explore the dependence of your Bayesian inferences on your priors. In particular, compare results with priors that are constant (e.g., as in part (d)) to those with logarithmic priors, i.e., $P_{\mathrm{pr}}(u_x) \sim u^{-1}$ and $P_{\mathrm{pr}}(u_y) \sim u^{-1}$.

(D)

$$\overbrace{P(U_x, U_y, g \mid R, h, g)}^{\text{Posterior}} = C\,\overbrace{P(U_x)P(U_y)\,P(g)}^{\text{Priors}}\ \overset{P(R,h,g)}{P(R,h,g \mid U_x, U_y, g)}$$

If priors are constants, then:

$$P(U_x, U_y, g \mid R, h, g) = C'\,P(R,h,g \mid U_x, U_y, g)$$
$$= C'\,G(R, R_0, \sigma_R)\,G(h, h_0, \sigma_h)\,G(g, g_0, \sigma_g)$$

$\mu(U_y) = 4.34937$ $\qquad \sigma_{U_y} = 0.454$

$\mu(U_x) = 11.215$ $\qquad \sigma_{U_x} = 1.15$

Through Gauss fit to posterior distributions

These values are very close to the frequentist inferences, but are slightly larger than in (a) and slightly lower than in (b). This is presumably due to the lack of a Jacobian in the Bayesian inference. This makes the posterior distribution simply the product of 3 Gaussians. The distribution for U_y is skewed left while the distribution for U_x is skewed right.

(E) W/ logarithmic priors,

$$P(U_x, U_y, g \mid R, h, g) = C\,\frac{1}{U_x}\,\frac{1}{U_y}\,P(R,h,g \mid U_x, U_y, g)$$

$\mu(U_y) = 4.349$ $\qquad \sigma_{U_y} = 0.454$

$\mu(U_x) = 11.210$ $\qquad \sigma_{U_x} = 1.15$

Though I include the priors in the marginalization, the numbers barely change. I would expect the distribution to be weighted toward small values of U_x and U_y. The posterior distribution is no longer simply the sum of 3 Gaussians.

For a given U_x, the difference between constant & log priors is ~ 0.002. The μ-values are in fact skewed a bit toward smaller values of U_x and U_y.

Figure 4 compares bayesian calculation, two kinds of priors give very close results. Figure **??** compares the difference in detail.

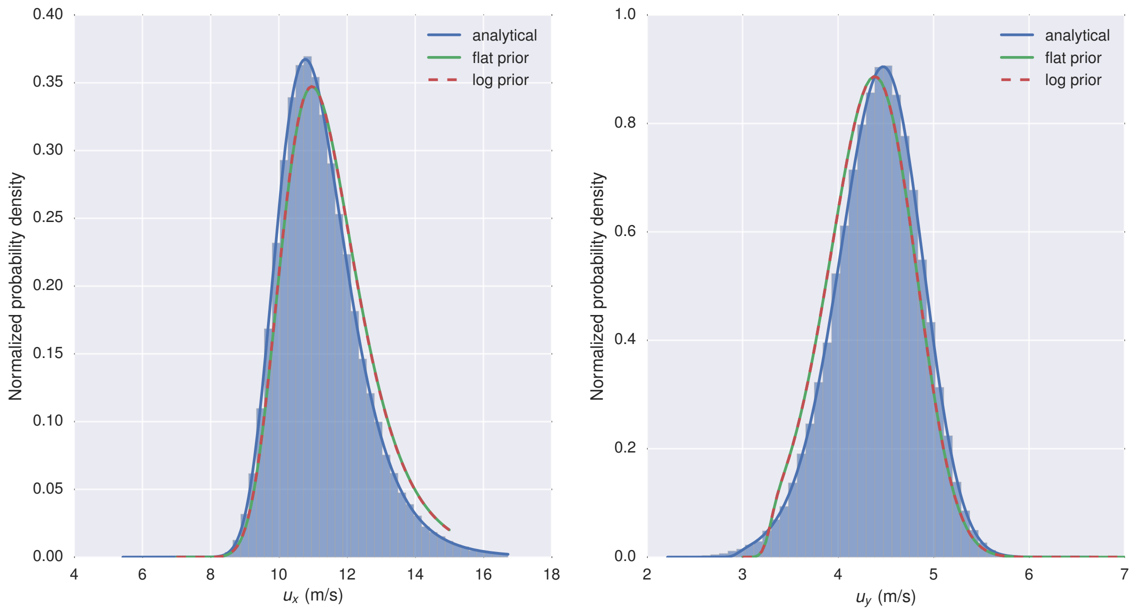

Figure 3: Comparison of Bayesian results and analytical results

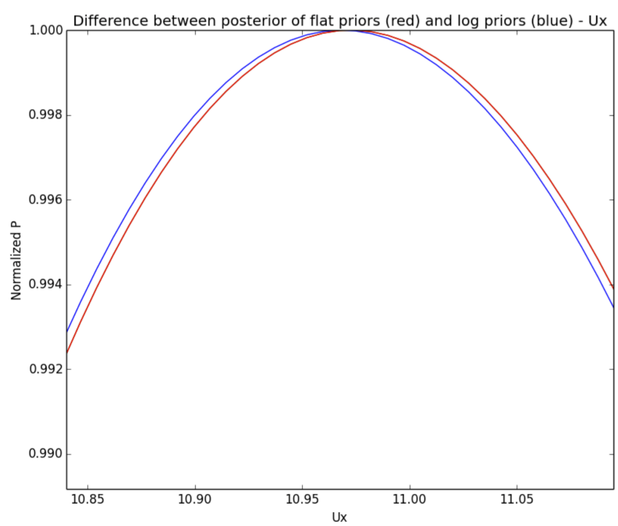

Figure 4: Comparison of affects of different priors on u_x

Code to compute the numerical integration.

```
import numpy as np
from scipy.integrate import dblquad
def dPux(g, uy, ux):
    h0 = 1.0
    dh = 0.2
    R0 = 10
    dR = 0.2
    g0 = 9.81
    dg = 0.05
    C = 1 / ((2 * np.pi)**(1.5)* dR * dh * dg) * 2 * uy**2 / g**2
    exp1 = np.exp(-(uy**2 / (2 * g) - h0)**2/(2 * dh**2))
    exp2 = np.exp(-(2 * ux * uy / g - R0)**2/(2 * dR**2))
    exp3 = np.exp(-(g - g0)**2/(2 * dg**2))
    return C * exp1 * exp2 * exp3

def Pux(ux):
    """
    numerical integration
    """
    # integration limit defined by 5 sigma limit
    return dblquad(lambda g, uy: dPux(g, uy, ux),
                   2.21, 6.64,
                   lambda g: 9.56,
                   lambda g: 10.06,
                   )

def Puy(uy):
    return dblquad(lambda g, ux: dPux(g, uy, ux),
                   5.42, 16.72,
                   lambda g: 9.56,
                   lambda g: 10.06,
                   )

nSigma = 5
ux = 11.074
dux = 1.130
uy = 4.429
duy = 0.443
ux0 = np.linspace(ux - nSigma*dux, ux + nSigma*dux, 200)
uy0 = np.linspace(uy - nSigma*duy, uy + nSigma*duy, 200)
PuxJco = np.array([Pux(uxi)[0] for uxi in ux0 ])
PuyJco = np.array([Puy(uyi)[0] for uyi in uy0 ])
```