Smith&Gelfand

April 10, 2024

Numerical example from Smith & Gelfand, 1992

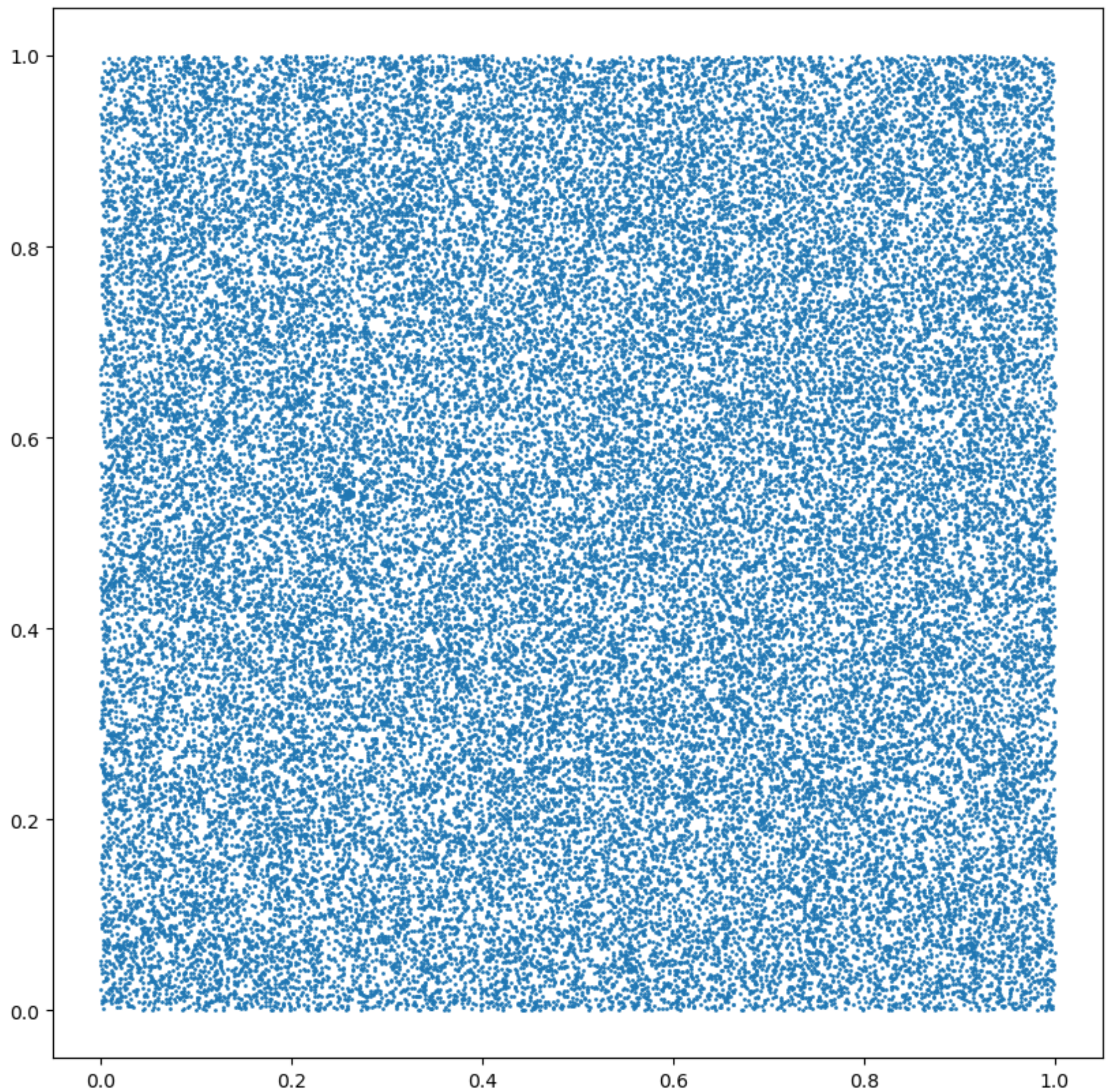
[20]:
```
import numpy as np
import matplotlib.pyplot as plt
rng = np.random.default_rng()
from scipy.special import binom, comb
```

[21]:
```
# data

n1=[5,6,4]
n2=[5,4,6]
yi=[7,5,6]
```

[22]:
```
# likelihood function

def likelihood(th1,th2):
    prod=1
    for i in range(3):
        tot=0
        for j in range(max(0,yi[i]-n2[i]),min(n1[i],yi[i])+1):
            tot += comb(n1[i],j,exact=True)*comb(n2[i],yi[i]-j,exact=True) *␣
 ↪th1**j *(1-th1)**(n1[i]-j) * th2**(yi[i]-j) * (1-th2)**(n2[i]-yi[i]+j)
        prod *= tot
    return prod
```

[23]:
```
# prior distribution (2D uniform distribution)

ns=50000
th1=np.array([rng.random() for i in range(ns)])
th2=np.array([rng.random() for i in range(ns)])
```

[24]:
```
plt.figure(figsize=(10,10))
plt.scatter(th1,th2,s=1)
plt.gca().set_aspect('equal')
plt.savefig('SGprior.pdf')
plt.show()
```

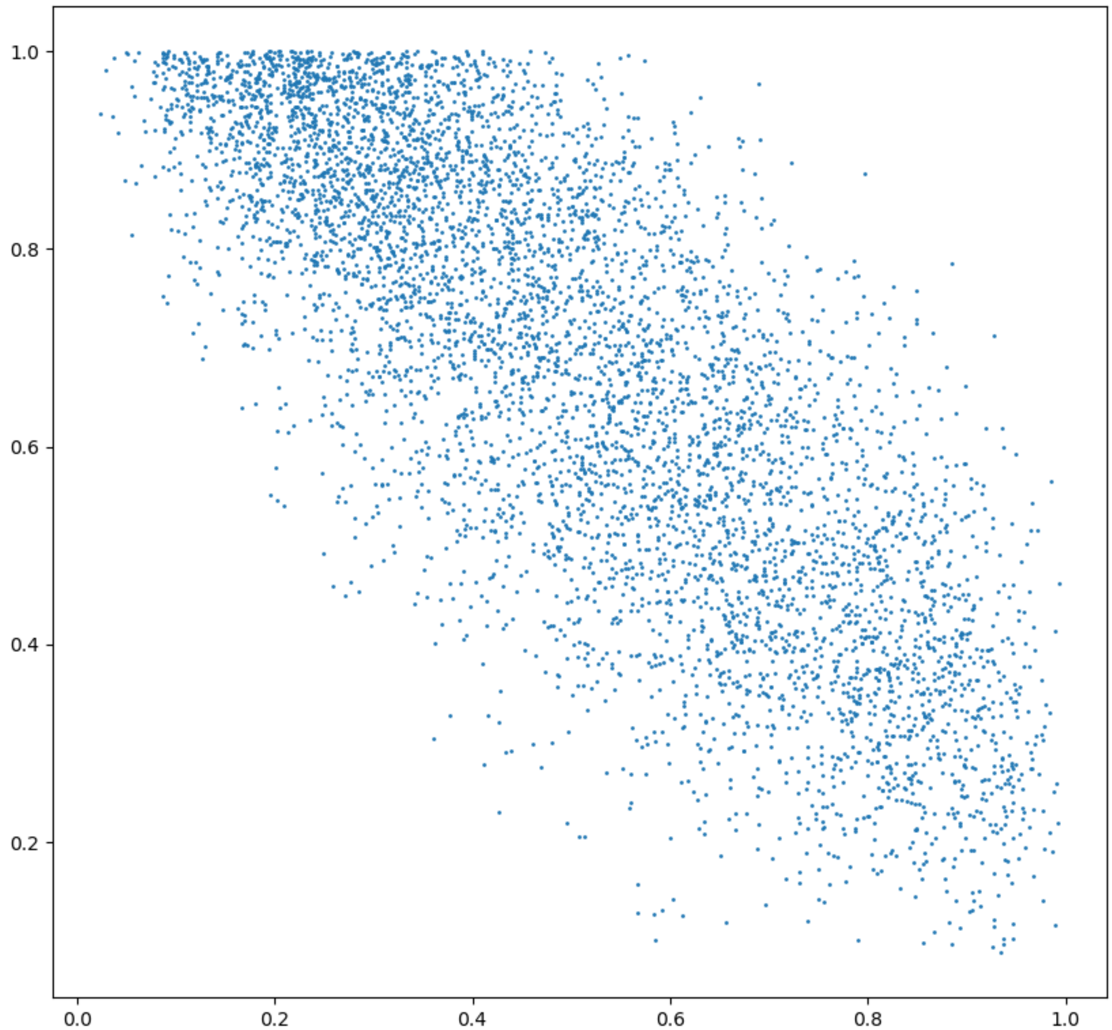


[25]:
```
# Posterior as a resampled prior using acceptance-rejection

w=np.array([likelihood(th1[i],th2[i]) for i in range(ns)])
maxl=max(w)
w /= maxl

th1w=np.zeros(ns)
th2w=np.zeros(ns)
ind=0
for i in range(ns):
    r=rng.random()
    if r < w[i]:
        th1w[ind]=th1[i]
```

```
        th2w[ind]=th2[i]
        ind += 1

print(f'{ind} accepted pairs (corresponding to {100*ind/float(ns)}% of total)')
```

5830 accepted pairs (corresponding to 11.66% of total)

[26]:
```
plt.figure(figsize=(10,10))
plt.scatter(th1w[0:ind],th2w[0:ind],s=1)
plt.gca().set_aspect('equal')
plt.savefig('SGacceptreject.pdf')
plt.show()
```

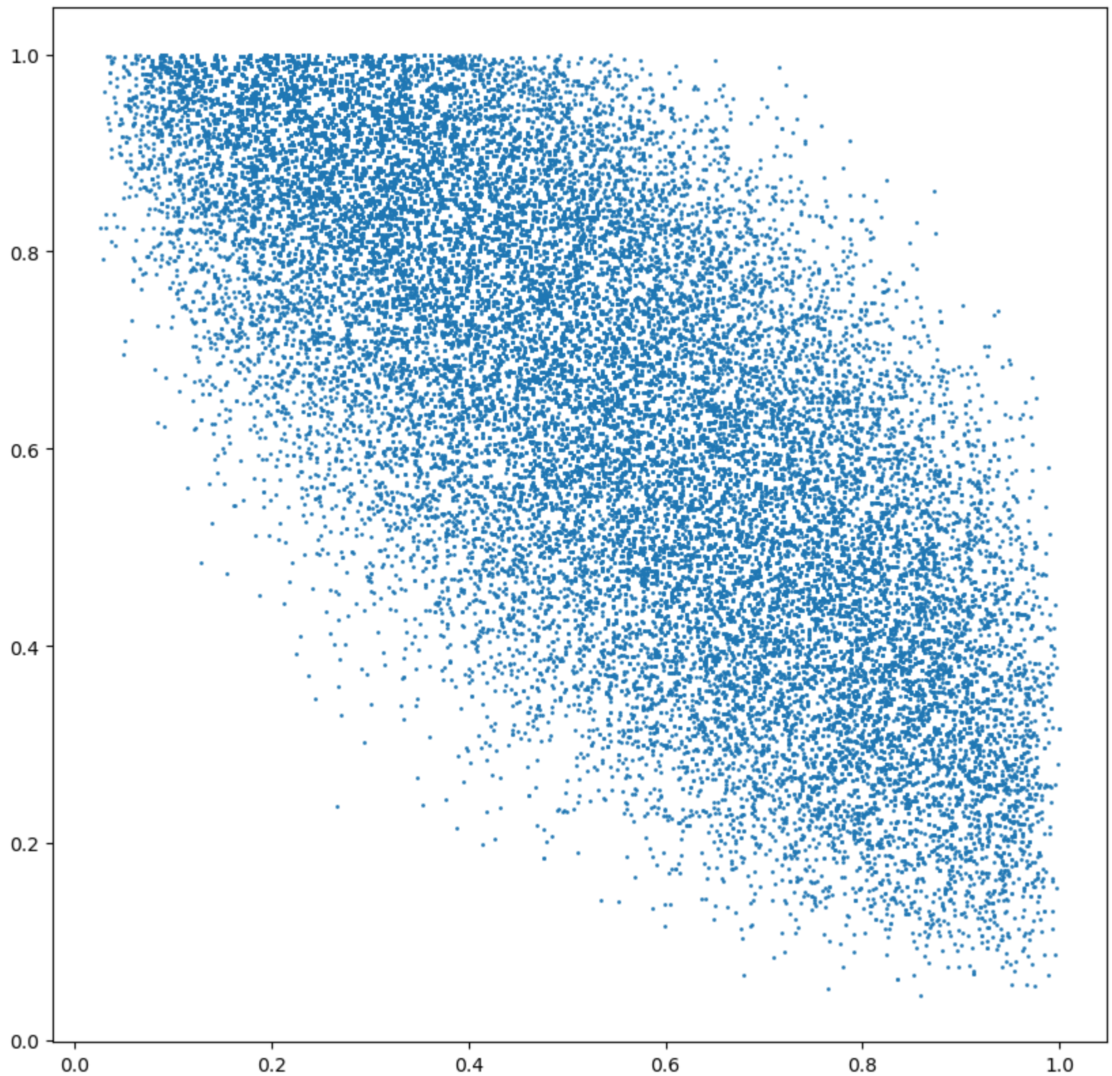


[27]:
```
# Posterior as a resampled prior using weighted bootstrap
```

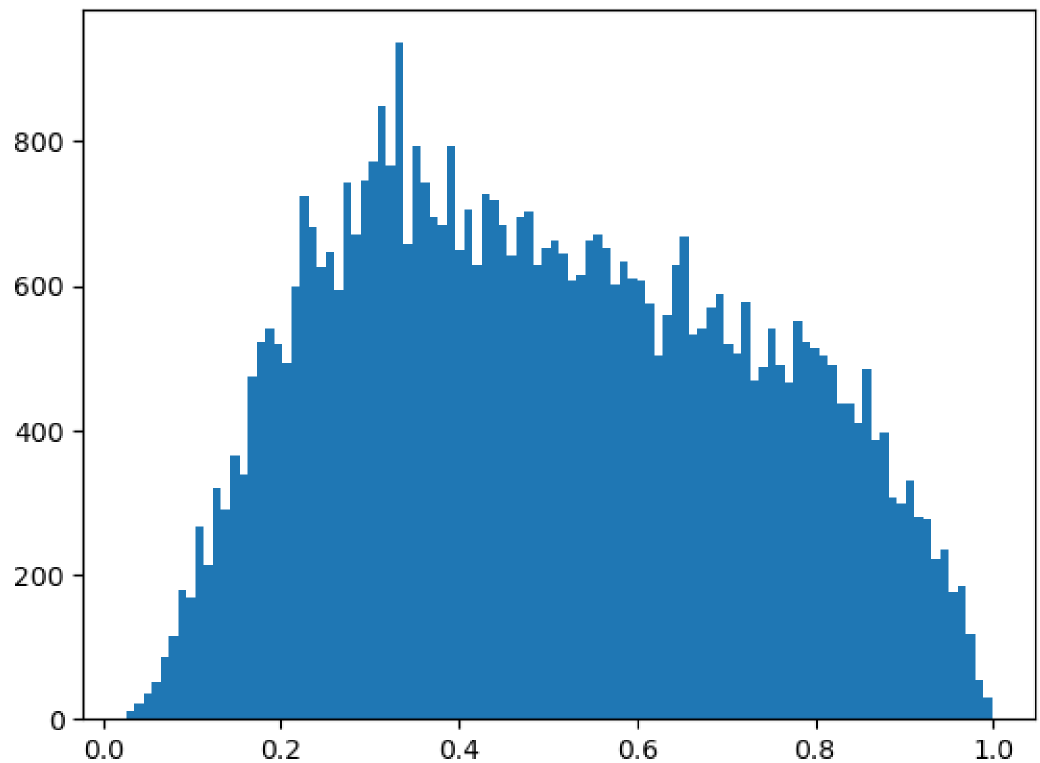
```
w=np.array([likelihood(th1[i],th2[i]) for i in range(ns)])
maxl=max(w)
w /= maxl

th1w=np.zeros(ns)
th2w=np.zeros(ns)
ind=0
while ind<ns:
    i=rng.integers(ns)
    r=rng.random()
    if r < w[i]:
        th1w[ind]=th1[i]
        th2w[ind]=th2[i]
        ind += 1

print(f'{ind} accepted pairs (corresponding to {100*ind/float(ns)}% of total)')
```

```
50000 accepted pairs (corresponding to 100.0% of total)
```

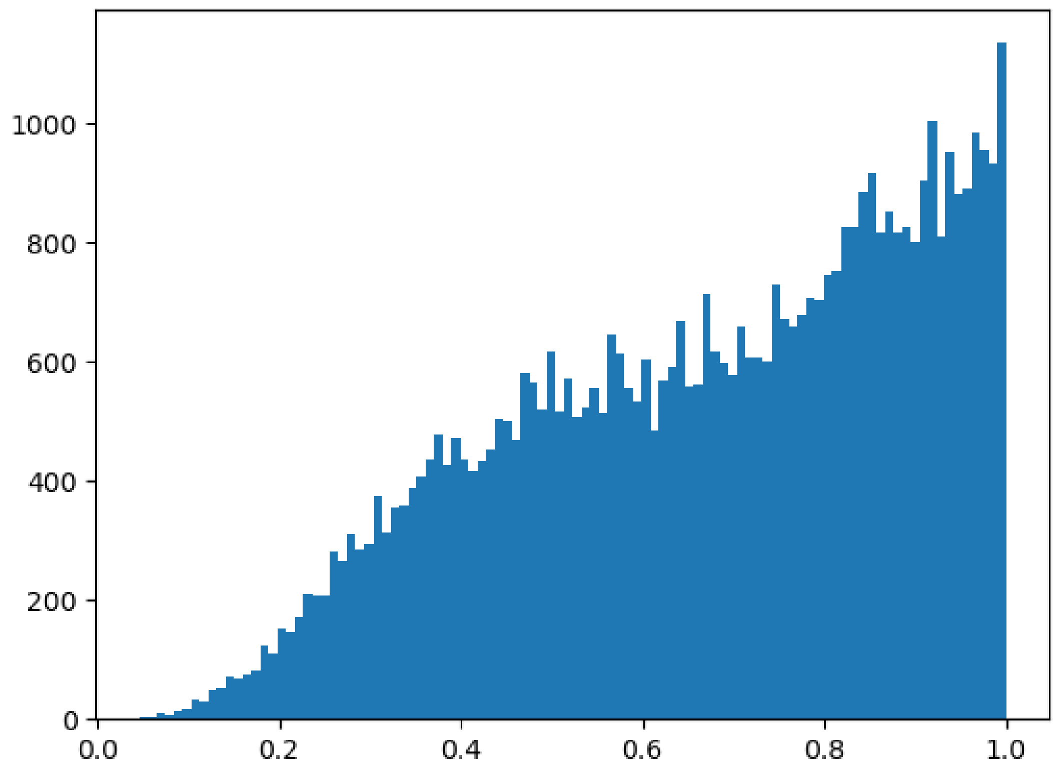
[28]:
```
plt.figure(figsize=(10,10))
plt.scatter(th1w,th2w,s=1)
plt.gca().set_aspect('equal')
plt.savefig('SGbootstrapped.pdf')
plt.show()
```



[29]:
```
# marginal distributions

plt.hist(th1w,100)
plt.savefig('th1w.pdf')
plt.show()
print(f'th1, mean = {np.mean(th1w)} ± {np.std(th1w)/np.sqrt(ns)}')
plt.hist(th2w,100)
plt.savefig('th2w.pdf')
plt.show()
print(f'th2, mean = {np.mean(th2w)} ± {np.std(th2w)/np.sqrt(ns)}')
```



`th1, mean = 0.502371939616515 ± 0.0010143863295156309`




```
th2, mean = 0.6747993750443025 ± 0.0010020381262689892
```

[ ]:

[ ]: