Approximate Bayesian Computing using Sequential Sampling

Jessi Cisewski
Carnegie Mellon University

June 2014

Recall: Basic ABC algorithm

For the observed data $y_{1:n}$, prior $\pi(\theta)$ and distance function ρ:

Algorithm

1. Sample θ^* from prior $\pi(\theta)$
2. Generate $x_{1:n}$ from forward process $f(y \mid \theta^*)$
3. Accept θ^* if $\rho(y_{1:n}, x_{1:n}) < \epsilon$
4. Return to step 1

Generates a sample from an approximation of the posterior:

$$f(x_{1:n} \mid \rho(y_{1:n}, x_{1:n}, \theta) < \epsilon) \cdot \pi(\theta) \approx f(y_{1:n} \mid \theta)\pi(\theta) \propto \pi(\theta \mid y_{1:n})$$

Summary of basic ABC

- Decisions that need to be made:
  1. Select distance function (ρ) and summary statistic(s)
  2. Tolerance (ϵ)
- Finding the "right" ϵ can be inefficient
  $\longrightarrow$ we end up throwing away many of the theories proposed from the selected priors
- How can we improve this basic algorithm?

Summary of basic ABC

- Decisions that need to be made:
  1. Select distance function (ρ) and summary statistic(s)
  2. Tolerance (ϵ)
- Finding the "right" ϵ can be inefficient
  $\longrightarrow$ we end up throwing away many of the theories proposed from the selected priors
- How can we improve this basic algorithm?

Example: Mean of a Gaussian with known σ^2

- Tolerance set to $\epsilon^* = 0.0234$
- Elapsed time to sample $N = 100$ particles: **14.189** seconds
- Total number of particles drawn: **46,120**

Summary of basic ABC

- Decisions that need to be made:
  1. Select distance function (ρ) and summary statistic(s)
  2. Tolerance (ϵ)
- Finding the "right" ϵ can be inefficient
  $\longrightarrow$ we end up throwing away many of the theories proposed from the selected priors
- How can we improve this basic algorithm?

Example: Mean of a Gaussian with known σ^2

- Tolerance set to $\epsilon^* = 0.0234$
- Elapsed time to sample $N = 100$ particles: **14.189** seconds
- Total number of particles drawn: **46,120**
- Sequentially (10 steps to ϵ^*): **0.499** seconds, **10,054** draws

Sequential ABC

Main idea

Instead of starting the ABC algorithm over with a smaller tolerance (ϵ), use the already sampled particle system as a proposal distribution *rather* than drawing from the prior distribution.

Sequential ABC

Main idea

Instead of starting the ABC algorithm over with a smaller tolerance (ϵ), use the already sampled particle system as a proposal distribution *rather* than drawing from the prior distribution.

Particle system: (1) retained sampled values, (2) importance weights

Decreasing tolerances $\epsilon_1 \geq \cdots \geq \epsilon_T$

ABC - Population Monte Carlo algorithm* (ABC - PMC)

1. At $t = 1$

   For $i = 1, \ldots, N$ particles

   Generate $\theta_i^{(1)} \sim \pi(\theta)$ and $x \sim f(y \mid \theta_i^{(1)})$ until $\rho(y, x) < \epsilon_1$

   Set $w_i^{(1)} = N^{-1}$

Decreasing tolerances $\epsilon_1 \geq \cdots \geq \epsilon_T$

ABC - Population Monte Carlo algorithm* (ABC - PMC)

1. At $t = 1$
   For $i = 1, \ldots, N$ particles

   Generate $\theta_i^{(1)} \sim \pi(\theta)$ and $x \sim f(y \mid \theta_i^{(1)})$ until $\rho(y, x) < \epsilon_1$
   Set $w_i^{(1)} = N^{-1}$
2. At $t = 2, \ldots, T$
   Set $\tau_t^2 = 2 \cdot \text{var}\left(\theta_{1:N}^{(t-1)}\right)$
   For $i = 1, \ldots, N$ particles

   Draw $\theta_i^* \sim \text{multinomial}\left(\theta_{1:N}^{(t-1)}, w_{1:N}^{(t-1)}\right)$
   Generate $\theta_i^{(t)} \mid \theta_i^* \sim N(\theta_i^*, \tau_t^2)$ and $x \sim f(y \mid \theta_i^{(t)})$ until $\rho(y, x) < \epsilon_t$
   Set $w_i^{(t)} \propto \pi(\theta_i^{(t)}) / \sum_{j=1}^N w_j^{(t-1)} \phi[\tau_t^{-1}(\theta_i^{(t)} - \theta_j^{(t-1)})]$

- $\phi(\cdot)$ is the density function of a $N(0, 1)$

*From Beaumont et al. (2009)

Recall: Mean of a Gaussian with known σ^2

Given the following model:

$$\mu \sim N(\mu_0, \sigma_0^2)$$
$$Y_i \mid \mu, \sigma^2 \sim N(\mu, \sigma^2)$$

The **posterior** is

$$\pi(\mu \mid y_{1:n}) \sim N(\mu_1, \sigma_1^2)$$

where

$$\mu_1 = \frac{\left(\frac{\mu_0}{\sigma_0^2} + \frac{\sum y_i}{\sigma^2}\right)}{\left(\frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}\right)}, \qquad \sigma_1^2 = \frac{1}{\left(\frac{1}{\sigma_0^2} + \frac{n}{\sigma^2}\right)}$$

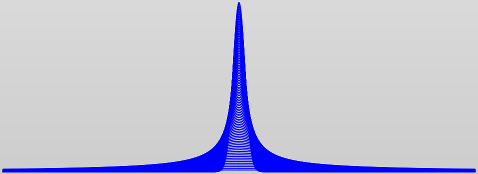
Recall: Mean of a Gaussian with σ^2 known: R code

```
n=25        #number of observations
N=1000      #particle sample size
true.mu = 0; sigma = 1
mu.hyper = 0; sigma.hyper = 10
data=rnorm(n,true.mu,sigma)
epsilon=0.005
mu=numeric(N)
rho=function(y,x) abs(sum(y)-sum(x))/n

for(i in 1:N){
d= epsilon +1
   while(d>epsilon) {
       proposed.mu=rnorm(1,0,sigma.hyper) #<--prior draw
       x=rnorm(n, proposed.mu, sigma)
       d=rho(data,x)}
mu[i]= proposed.mu}}
```

Mean of a Gaussian with σ^2 known

Sequential version



Mean of a Gaussian with σ^2 known: Sequential R code

```
# INPUTS
n=25  #number of observations
N=2500  #particle sample size
true.mu = 0
sigma = 1
mu.hyper = 0
sigma.hyper = 10
data=rnorm(n,true.mu,sigma)
epsilon = 1
time.steps = 20
weights = matrix(1/N,time.steps,N)
mu=matrix(NA,time.steps,N)
d=matrix(NA,time.steps,N)
rho=function(y,x) abs(sum(y)-sum(x))/n
```

Mean of a Gaussian with σ^2 known: Sequential R code

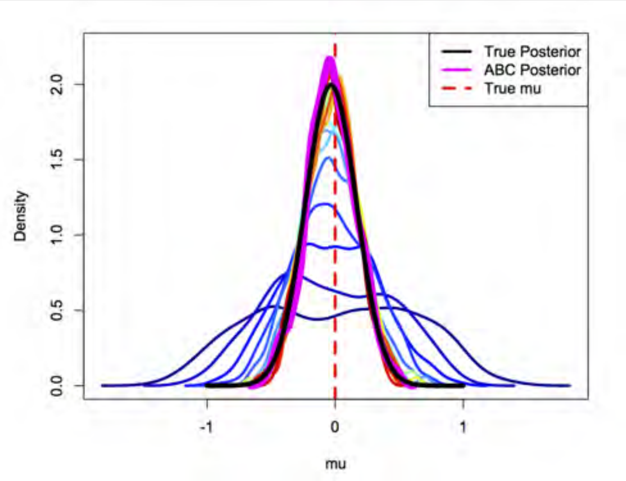
```
for(t in 1:time.steps){
   if(t==1){
      for(i in 1:N){
         d[t,i]= epsilon +1
         while(d[t,i]>epsilon) {
            proposed.mu=rnorm(1,0,sigma.hyper) #<--prior draw
            x=rnorm(n, proposed.mu, sigma)
            d[t,i]=rho(data,x)}  mu[t,i]= proposed.mu
   }} else{[NEXT SLIDE]}
}
```

Mean of a Gaussian with σ^2 known: Sequential R code

```
for(t in 1:time.steps){ if(t==1){[PREVIOUS SLIDE]} else{
   epsilon = c(epsilon,quantile(d[t-1,],.75))
   mean.prev <- sum(mu[t-1,]*weights[t-1,])
   var.prev <- sum((mu[t-1,] - mean.prev)^2*weights[t-1,])
   for(i in 1:N){d[t,i]= epsilon[t]+1
      while(d[t,i]>epsilon[t]) {
      sample.particle <- sample(N, 1, prob = weights[t-1,])
      proposed.mu0 <- mu[t-1, sample.particle]
      proposed.mu <- rnorm(1, proposed.mu0, sqrt(2*var.prev))
      x <- matrix(rnorm(n,proposed.mu, sigma),n,1)
      d[t,i]=rho(data,x) }
   mu[t,i]= proposed.mu
   mu.weights.denominator<-
        sum(weights[t-1,]*dnorm(proposed.mu,mu[t-1,],sqrt(2*var.prev)))
   mu.weights.numerator<-dnorm(proposed.mu,0,sigma.hyper)
   weights[t,i] <- mu.weights.numerator/mu.weights.denominator
   }}
weights[t,] <- weights[t,]/sum(weights[t,])}
```

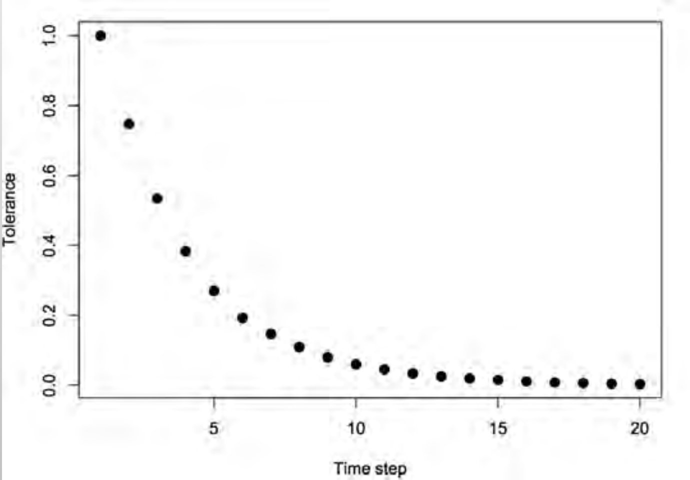
Mean of a Gaussian with σ^2 known: Sequential

$N = 1000$, $n = 25$



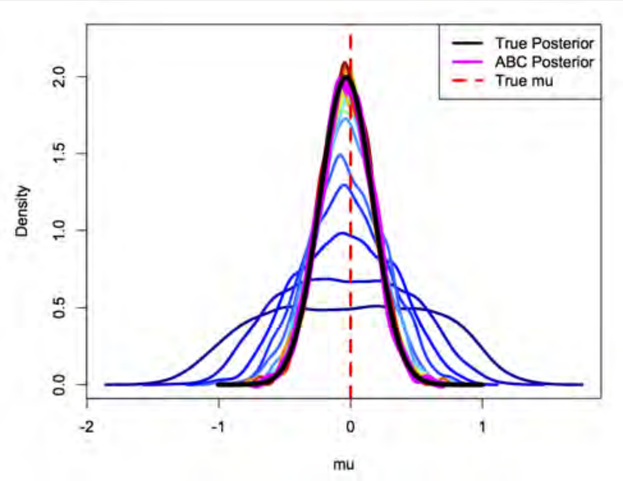
Mean of a Gaussian with σ^2 known: Sequential

$N = 1000,\ n = 25$



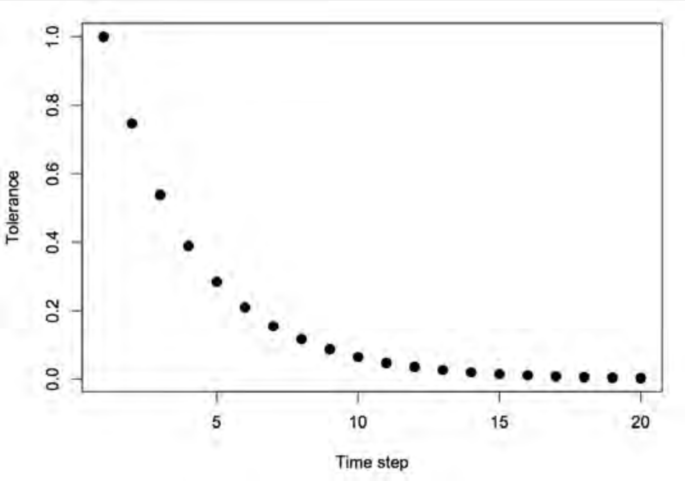
Mean of a Gaussian with σ^2 known: Sequential

$N = 2500$, $n = 25$



Mean of a Gaussian with σ^2 known: Sequential

$N = 2500,\ n = 25$



Sequential setting: decisions

1. Determining the sequence of tolerances, $\epsilon_{1:t}$
2. Moving the particles between time steps
3. Calculating the particle weights

Sequential ABC for Type Ia Supernovae

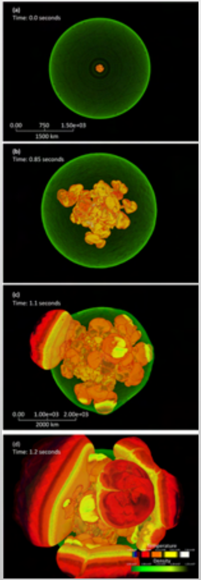


Image: Argonne National Laboratory

Recall: ABC for Type Ia Supernovae

Goal: posteriors for Ω_M and ω

Initial ABC Algorithm steps

1. Simulate from priors: $\Omega_M^* \sim U(0,1)$ and $\omega^* \sim U(-3,0)$
   $z^* \sim (1+z)^\beta$, $\beta = 1.5 \pm 0.6$
2. Obtain sample of μ^* via Ia light curve generating forward model $f(\Omega_M^*, \omega^*, z^*, \eta)$
3. Nonparametric smoothing of generated sample (z^*, μ^*): $(\tilde{z}, \tilde{\mu})$
4. If $\rho((\tilde{z}, \tilde{\mu}), (z, \mu)) \leq \epsilon \longrightarrow$ keep Ω_M^* and ω^*

- η = nuisance parameters, (z, μ) are the *smoothed* real observations
- Forward model uses SNANA/MLCS2k2 to get the Ia SN light curves

⋆ Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential

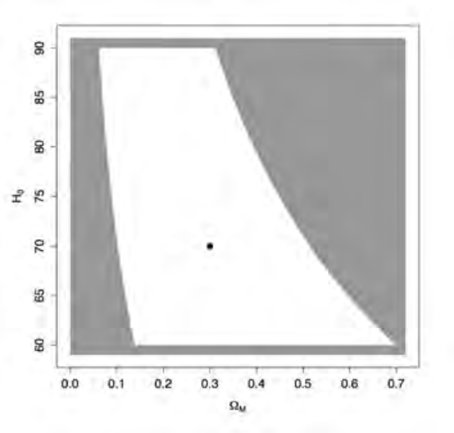


Figure: Chad Schafer

⋆ Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential

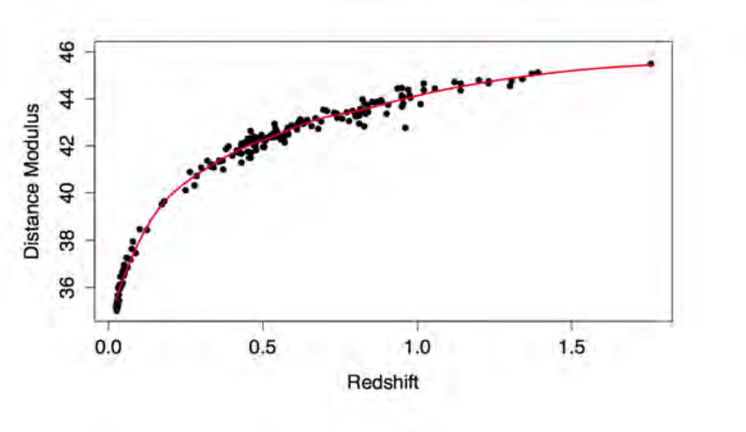


Figure: Chad Schafer

★ Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential

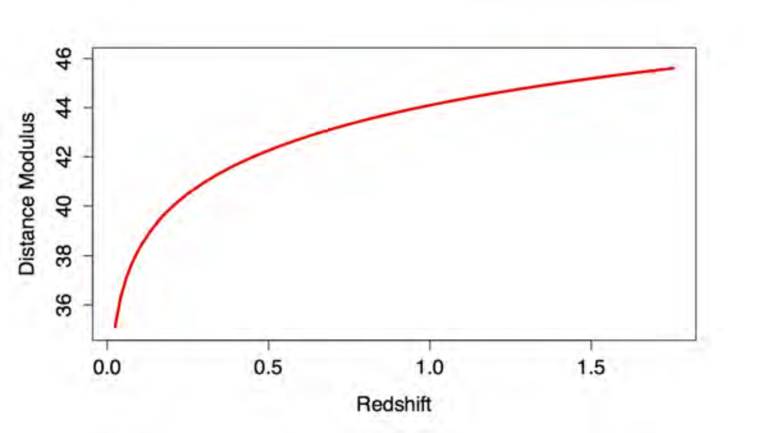


Figure: Chad Schafer

ABC for Type Ia Supernovae: sequential

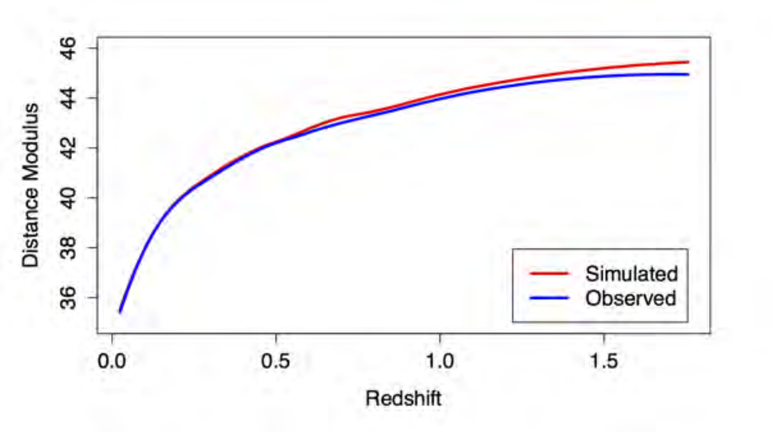


Figure: Chad Schafer

ABC for Type Ia Supernovae: sequential

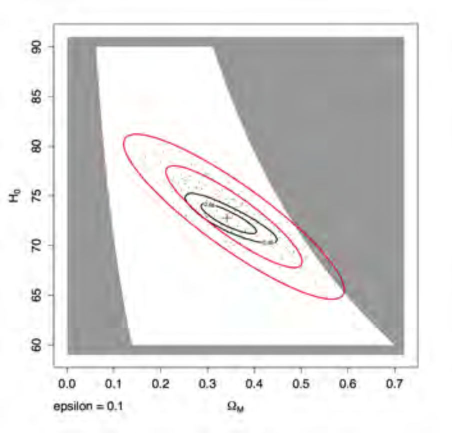


Figure: Chad Schafer

★ Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential

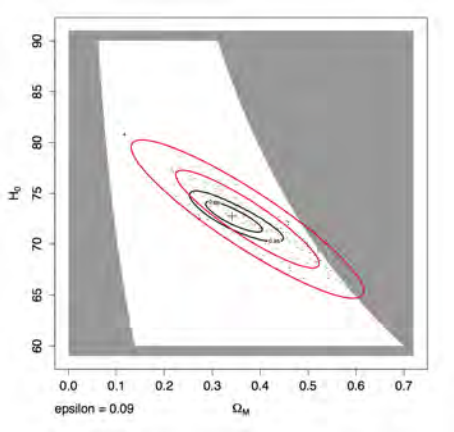


Figure: Chad Schafer

- Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential

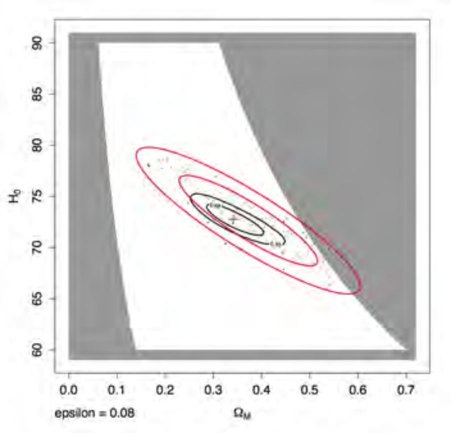


Figure: Chad Schafer

★ Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential

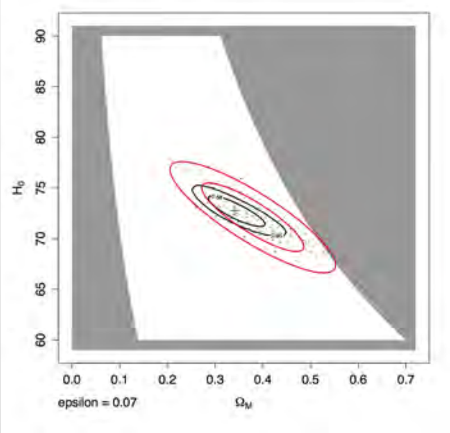


Figure: Chad Schafer

★ Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential

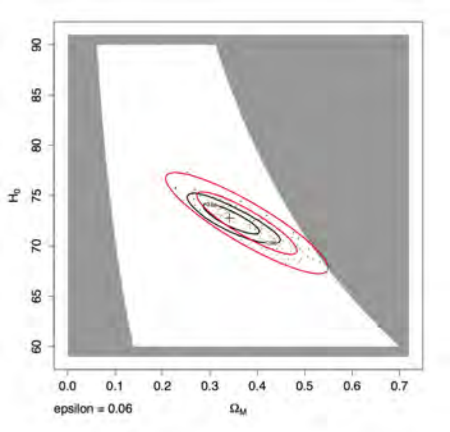


Figure: Chad Schafer

- ⋆ Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential

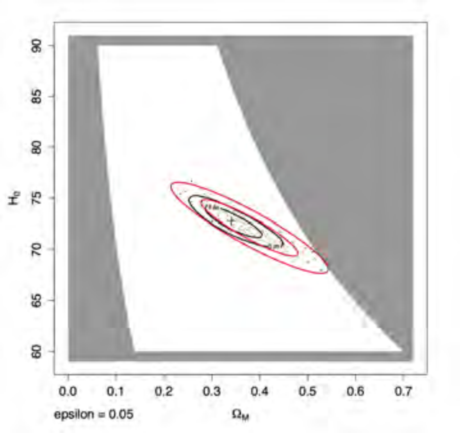


Figure: Chad Schafer

★ Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential

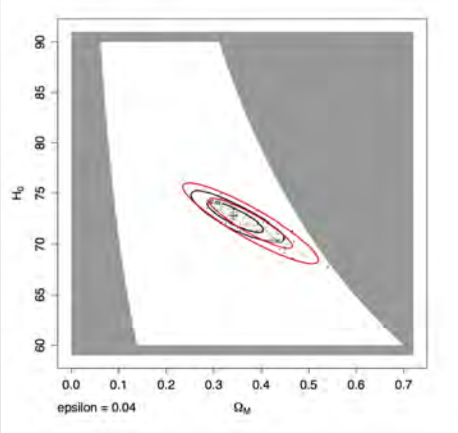


Figure: Chad Schafer

⋆ Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential

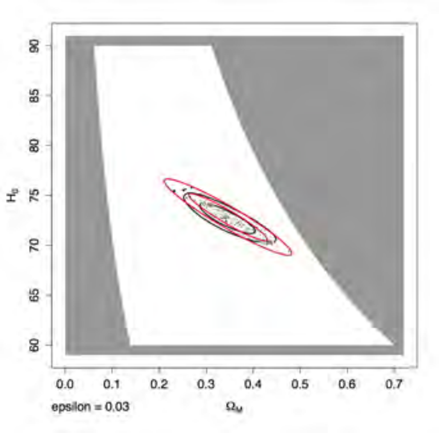


Figure: Chad Schafer

⋆ Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential

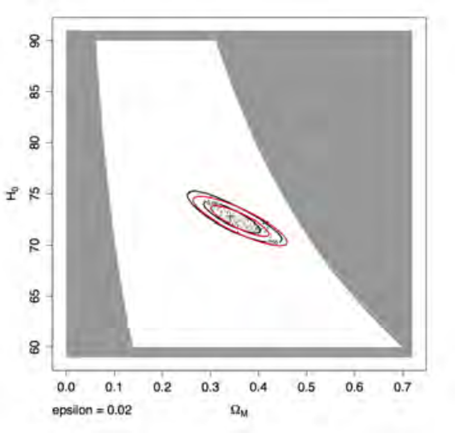


Figure: Chad Schafer

★ Weyant et al. (2013)

ABC for Type Ia Supernovae: sequential details

- Tolerance, ϵ_t, selected selected from distribution of $\rho_{t-1}^{(J)} = \rho\left((\tilde{z}, \tilde{\mu}), (z, \mu)\right) \leq \epsilon_{t-1},\ J = 1, \ldots, N$

  At $t = 1$, keep all points

  At $t = 2$, $\epsilon_2 = 25th$ percentile of sample of $\{\rho_1^{(J)}\}_{J=1}^N$

  For $t \geq 3$, $\epsilon_t = 50th$ percentile of sample of $\{\rho_{t-1}^{(J)}\}_{J=1}^N$

⋆ Weyant et al. (2013)

Sequential ABC for the Stellar IMF

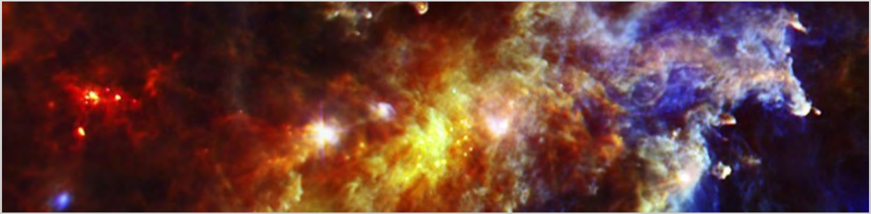
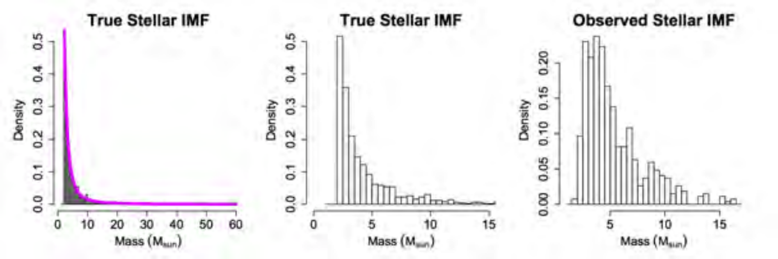


Image: https://astrojournalclub.files.wordpress.com/2011/06/cropped-rosette_herschel_hi.jpg

Stellar IMF

$$f_M(x \mid \alpha, M_{\max}) = cx^{-\alpha}, \quad x \in [M_{\min}, M_{\max}]$$



- Aged 10 Myrs according to $M_{cutoff} = \text{Age}^{-2/5} \times 10^{8/5}$
- Uncertainty: $\log m_i = \log M_i + \sigma_i \eta_i$ (with $\eta_i \sim N(0,1)$)
- Observational completeness:

$$P(obs \mid m) = \begin{cases} 0, & m < C_{min} \\ \frac{m - C_{min}}{C_{max} - C_{min}}, & m \in [C_{min}, C_{max}] \\ 1, & m > C_{max}. \end{cases}$$

Stellar IMF: Sequential ABC algorithm

Data: Observed stellar masses
Result: ABC-posterior sample of θ
At iteration $t = 1$ *for each of* $j = 1, \ldots, N$:
while $\rho(m_{sim}, m_{obs}) > \epsilon_t$ **do**
    Propose $\theta_t^{(j)}$ by drawing $\theta_t^* \sim p(\theta)$
    Generate cluster stellar masses m_{sim} from $f(x \mid \theta_t^*)$
    Calculate distance $\rho(m_{sim}, m_{obs})$
end
$\theta_t^{(J)} \leftarrow \theta_t^*$
$W_t^{(j)} \leftarrow 1/N$
At iterations $t = 2, \ldots, T$:
for $j = 1, \ldots, N$ **do**
    while $\rho(m_{sim}, m_{obs}) > \epsilon_t$ **do**
        Select $\theta^{(j)}$ by drawing from the $\theta_{t-1}^{(i)}$ with probabilities $W_{t-1}^{(i)}$
        Generate $\theta^{*(j)}$ from transition kernel $K(\theta^{(j)}, \cdot)$
        Generate cluster stellar masses m_{sim} from $f(x \mid \theta^{*(j)})$
        Calculate distance $\rho(m_{sim}, m_{obs})$
    end
    $\theta_t^{(j)} \leftarrow \theta^{*(j)}$
    $W_t^{(j)} \leftarrow \frac{p(\theta_t^{(j)})}{\sum_{i=1}^{N} W_{t-1}^{(i)} K(\theta_{t-1}^{(i)}, \theta_t^{(j)})}$
end

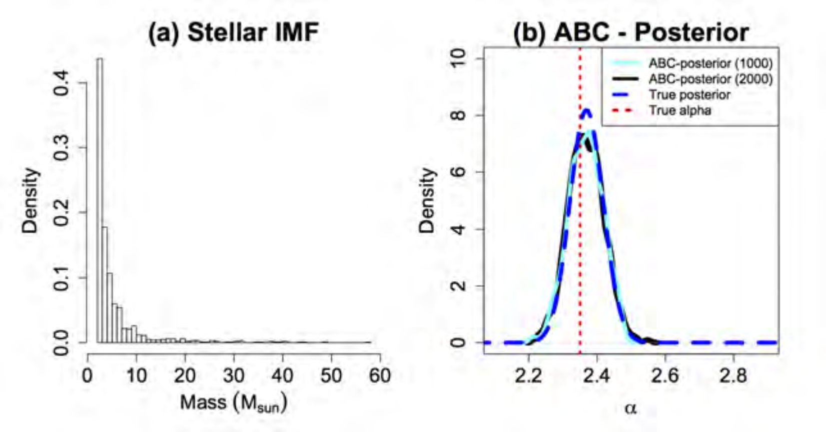
Image: Weller et al. (2014)

Stellar IMF: Sequential ABC summary

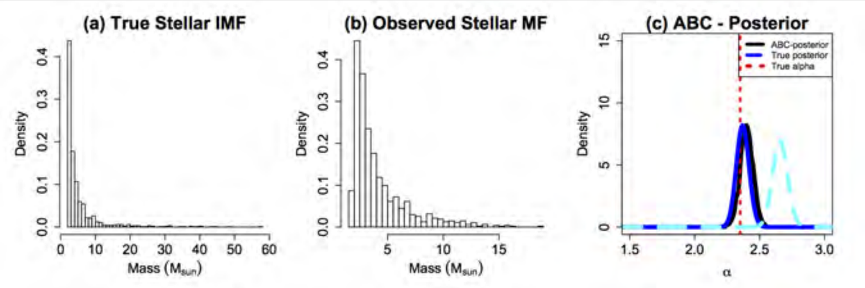
- Summary statistic: smooth IMF on log-masses, number of stars observed
- Distance function:

$$\rho(m_{sim}, m_{obs}) = \left(\left[\int \left\{ \hat{f}_{\log m_{sim}}(x) - \hat{f}_{\log m_{obs}}(x) \right\}^2 dx \right]^{1/2}, \left| 1 - \frac{n_{sim}}{n_{obs}} \right| \right)$$

- Tolerance: sequential (decrease based on previous time step's retained distances)
- Transition kernel: multivariate Gaussian



(a) Stellar IMF
Density
0.0
0.1
0.2
0.3
0.4
0
10
20
30
40
50
60
Mass (M_sun)
(b) ABC - Posterior
Density
0
2
4
6
8
10
2.2
2.4
2.6
2.8
α
ABC-posterior (1000)
ABC-posterior (2000)
True posterior
True alpha



The light blue curve in Figure (c) is the true posterior of the *observed* MF, but we want the posterior of *true* IMF.

Concluding remarks

1. Approximate Bayesian Computation could be a useful tool in astronomy, but it must be handled with care
2. There are three main decisions that need to be made in the standard ABC algorithm: summary statistic, distance function, and tolerance
3. Considering a sequence of tolerances can lead to more efficient sampling, but results in more decisions: how to decrease the tolerance, when to stop the sampling, how to "move" or "mix" the particles between sampling steps

Bibliography

Beaumont, M. A., Cornuet, J.-M., Marin, J.-M., and Robert, C. P. (2009), "Adaptive approximate Bayesian computation," *Biometrika*, 96, 983 – 990.

Weller, G. B., Cisewski, J., Schafer, C. M., and Hogg, D. W. (2014), "Approximate Bayesian Computation for the High Mass End of the Stellar Initial Mass Function," In preparation.

Weyant, A., Schafer, C., and Wood-Vasey, W. M. (2013), "Likeihood-free cosmological inference with type Ia supernovae: approximate Bayesian computation for a complete treatment of uncertainty," *The Astrophysical Journal*, 764, 116.