Confidence intervals for prevalence

Contents

Introduction

Here I try to explain some of the errors that must have been made in the preprint: https://www.medrxiv.org/content/10.1101/2020.04.14.20062463v1.full.pdf in estimating the 95% CIs of the true community prevalence of COVID-19 from their data. For clarity, I will ignore issues in the way they chose their sample and also their post-stratification adjustments. I am also ignoring effects of non-perfect sensitivity of the tests, for now. This purely focuses on the effects of the uncertainty in the estimates of false-positive rate / sensitivity on the uncertainty in the estimates of prevalence, which they appear to have ignored.

Estimation of false-positive rate / specificity of their test

As background, I will first discuss the estimation of the false-positive rate and its 95% CI.

In medicine it is common to quote the specificity of a given test, meaning the fraction of tests of known true negatives that are properly classified as negative by the test. I will instead work with the false-positive rate, which is just 1 - specificity, and denote it by ρ.

To estimate the false-positive rate, the test is run on a set of known negative samples and the number of resulting positives is noted. Denote the total number of such tests run as n and the number of positives as m. The preprint references two such experiments, one with $m = 0$, $n = 30$, and another with $m = 2$, $n = 371$. It would also be reasonable to consider these two experiments as one big experiment with $m = 2$, $n = 401$.

Running a single such test is an example of a Bernoulli trial where the probability of a positive result is ρ. There are a number of ways to estimate ρ from a series of such trials. Here I will use a Bayesian framework. It is very simple in this case: the likelihood has a factor of ρ for every positive test result and a factor of $1 - \rho$ for every negative test result. So using a uniform prior, the pdf of the posterior distribution will just be proportional to $\rho^m(1-\rho)^{n-m}$. This is a beta distribution (https://en.wikipedia.org/wiki/Beta_distribution). Use $\beta(x; a, b)$ to denote the pdf of a beta distribution with shape parameters a and b:

$$\beta(x; a, b) = \frac{x^{a-1}(1-x)^{b-1}}{B(a, b)},$$

where $B(a, b)$ is the complete beta function. Then the pdf of our posterior distribution is

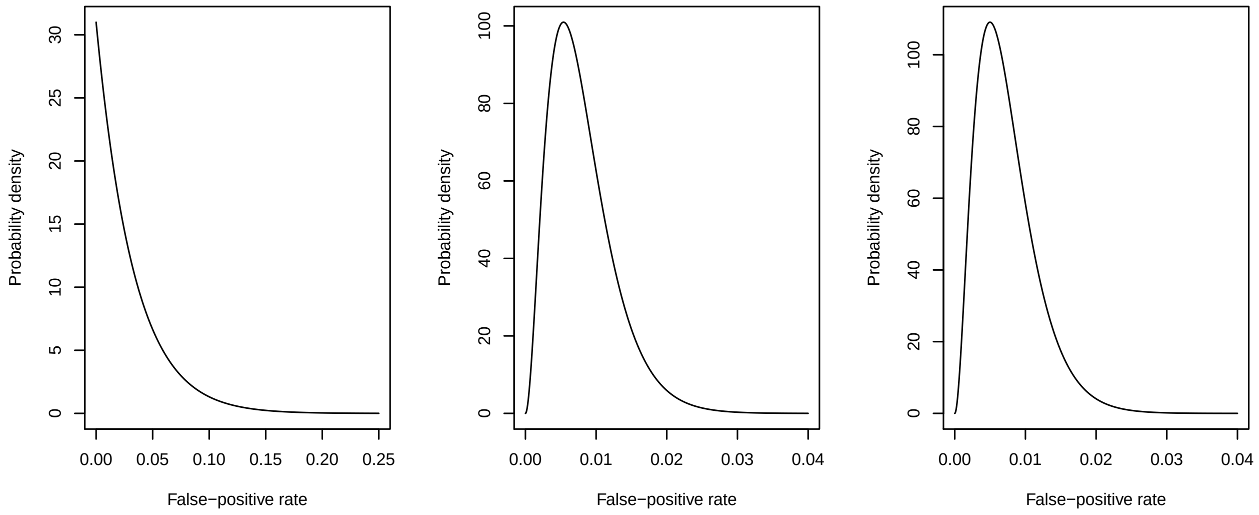
$$\phi_\rho(\rho) = \beta(\rho; m+1, n-m+1).$$

R has builtin functions for the pdf, cdf, quantiles, etc. of beta distributions (`dbeta`, `pbeta`, and `qbeta`, respectively).

Let's plot the pdfs and calculate point estimates and 95% CIs for the three m, n pairs relevant here. For the point estimate, I'll use both the mean of the posterior pdf ($a/(a+b) = (m+1)/(n+2)$) and the mode/maximum likelihood ($(a-1)/(a+b-2) = m/n$). For the 95% CI I'll use the central 95% of the distribution (*i.e.* range between quantiles 0.025 and 0.975), unless the mode of the pdf is at the lower boundary of the domain (that is, 0) in which case I'll use the range between quantiles 0 and 0.95.

Note the x-axis limits are different in the first plot from the other two.

```r
par(mfrow=c(1,3))

rholim = c(0, 0.25)
nrho = 999
rho = seq(rholim[1], rholim[2], length=nrho)

m = 0; n = 30
plot(rho, dbeta(rho, m+1, n-m+1), typ='l', xlab="False-positive rate",
     ylab="Probability density", ylim=c(0, max(dbeta(rho, m+1, n-m+1))))
meanrho1 = (m+1)/(n+2)
moderho1 = m/n
rhoci1 = qbeta(c(0.0, 0.95), m+1, n-m+1)


rholim = c(0, 0.04)
rho = seq(rholim[1], rholim[2], length=nrho)
m = 2; n = 371
plot(rho, dbeta(rho, m+1, n-m+1), typ='l', xlab="False-positive rate",
     ylab="Probability density")
meanrho2 = (m+1)/(n+2)
moderho2 = m/n
rhoci2 = qbeta(c(0.025, 0.975), m+1, n-m+1)

m = 2; n = 401
plot(rho, dbeta(rho, m+1, n-m+1), typ='l', xlab="False-positive rate",
     ylab="Probability density")
```



```
meanrho3 = (m+1)/(n+2)
moderho3 = m/n
rhoci3 = qbeta(c(0.025, 0.975), m+1, n-m+1)
```

The point estimates and 95% CIs of the false-positive rate, ρ, are

	mean	mode	95% CI
$m = 0, n = 30$	0.031	0	0, 0.092
$m = 2, n = 371$	0.008	0.0054	0.0017, 0.019
$m = 2, n = 401$	0.0074	0.005	0.0015, 0.018

For comparision with their stated results near the end of the first paragraph of their Results section, I'll convert to these to specificity percentages:

	mean	mode	95% CI
$m = 0, n = 30$	96.9	100	90.8, 100
$m = 2, n = 371$	99.2	99.5	98.1, 99.8
$m = 2, n = 401$	99.3	99.5	98.2, 99.8

The preprint gives

	point estimate	95% CI
$m = 0, n = 30$	100	90.5, 100
$m = 2, n = 371$	99.5	98.1, 99.9
$m = 2, n = 401$	99.5	98.3, 99.9

Their point estimates agree well with the Bayesian modal point estimates and the 95% CIs agree quite well also. Their problem is that they didn't take these into account in the estimates of the 95% CIs of the true prevalence in their sample, as we'll see below.

Estimates of the true prevalence

First, as a sanity check, note that in their testing they got $M = 50$ positive results from $N = 3330$ tests. If the false-positive rate were zero, this would indicate a prevalence of around $50/3330 \approx 0.015$. But note that *all* of the 95% CIs for the false-positive rate go above 0.015. This means that it is a possibility (at the greater than 5% level) that all or nearly all of their positive tests were false-positives, and thus their 95% CIs for the (unadjusted) prevalence must include or nearly include zero (in which case their 95% CIs for adjusted prevalence should also). The fact that their 95% CIs do not go down to (near) zero indicates they have made a basic mistake.

In a Bayesian framework, it is again easy to calculate a posterior distribution and from that point estimates and CIs. Ignoring, for now, the false-negative rate (that is, assuming it is zero / the sensitivity is 1), the probability of getting a positive result from a test with a false-positive rate of ρ in a population with a prevalence of P is just $P + \rho(1-P) = P(1-\rho) + \rho$. That just says that all the true positives return a positive results plus a fraction ρ of the true negatives also do. And so the probability of getting a negative test result is just that subtracted from 1: $1 - P(1-\rho) - \rho$.

So, after getting m positive results from n tests of known negative patients and M positive results from N tests from a sample of the population whose prevalence is P, our posterior pdf (again assuming a uniform prior) for the joint distribution of ρ and P is propotional to all the factors from each test:

$$\phi_{P,\rho} \propto \rho^m (1-\rho)^{n-m} \left(P(1-\rho)+\rho\right)^M \left(1 - P(1-\rho) - \rho\right)^{N-M}$$

I don't believe that this distribution has a common name and therefore builtin functions for calculating the normalized version of it, its cdf, quantiles, etc. don't exist, so I'll just do all those calculations below numerically. (It might be possible to write it in terms of beta distributions?).

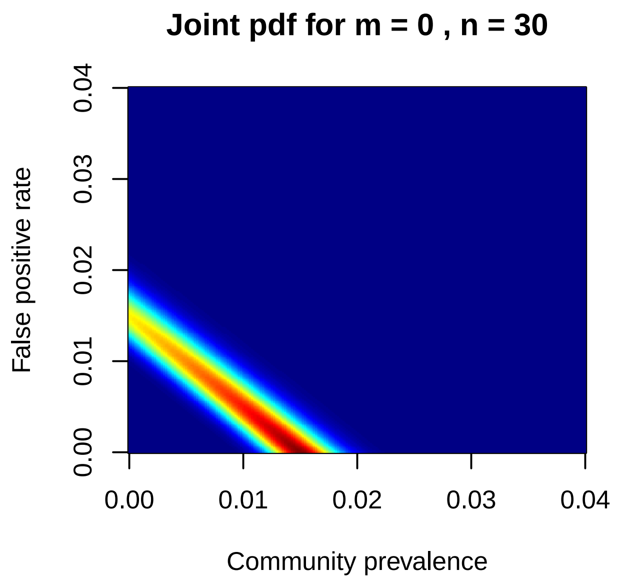
```r
# note: unnormalized
jointlikelihood = function(P, rho, m, n, M, N) {
    rho^m * (1-rho)^(n-m) * (P*(1-rho) + rho)^M * (1 - P*(1-rho) - rho)^(N-M)
}
```

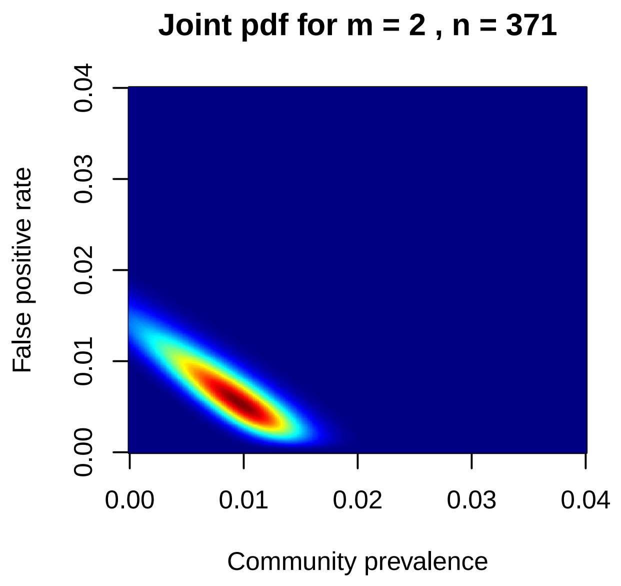
Joint pdfs

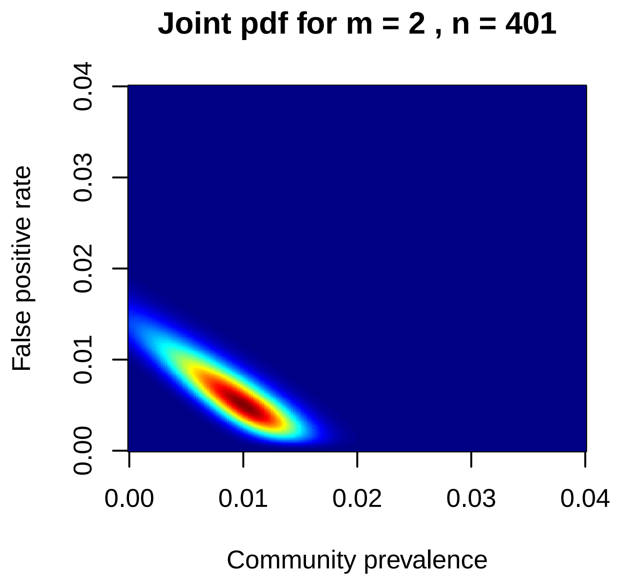
First let's plot the joint pdfs for the three false-positive m and n.

```r
par(mfrow=c(2,2))

plotpdf = function(m, n, M, N, nP=250, nrho=249, Plim=c(0, 0.04), rholim=c(0, 0.04)) {
  rho = seq(rholim[1], rholim[2], length=nrho)
  P = seq(Plim[1], Plim[2], length=nP)
  j = outer(P, rho, jointlikelihood, m=m, n=n, M=M, N=N)
  j = j / sum(j * mean(diff(P)) * mean(diff(rho)))

  image(P, rho, j, col=jet(100), xlab="Community prevalence", ylab="False positive rate",
        main=paste("Joint pdf for m =", m, ", n =", n))

}

M = 50
N = 3330

m = 0; n = 30
plotpdf(m, n, M, N)
```

```r
m = 2; n = 371
plotpdf(m, n, M, N)
m = 2; n = 401
plotpdf(m, n, M, N)
```







So basically all the data (the known negative tests and the community tests) are consistent with the false-positive rate being something like 1.5% and the community prevalence being very low, or vice-versa, or roughly a linear combination thereof.

Another interesting thing is that while the 95% CI for ρ went up to around 10% from the $m = 0$, $n = 30$ specificity test, performing 3330 community tests and only having 1.5% of them come back positive has more tightly constrained the false-positive rate.

Estimates of the community prevalence

From marginal distributions (the right way)

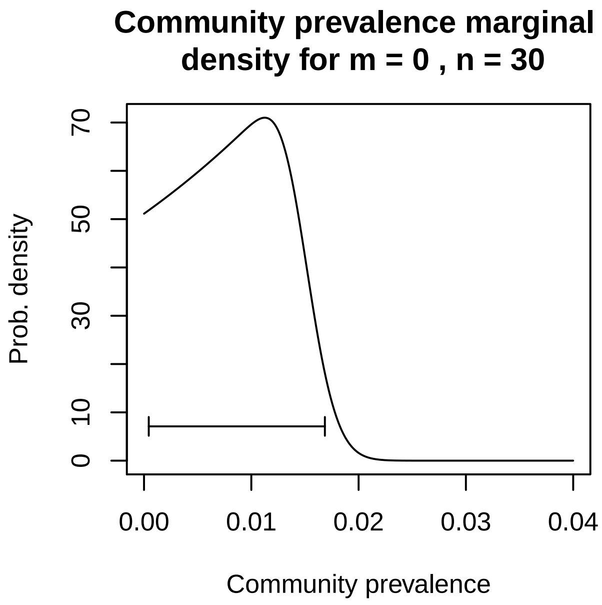
Estimates and 95% CIs of the community prevalence come from the marginal distribution of P. That is, we don't know the true false-positive rate ρ, so we integrate over all possible ρ weighted by the joint pdf:

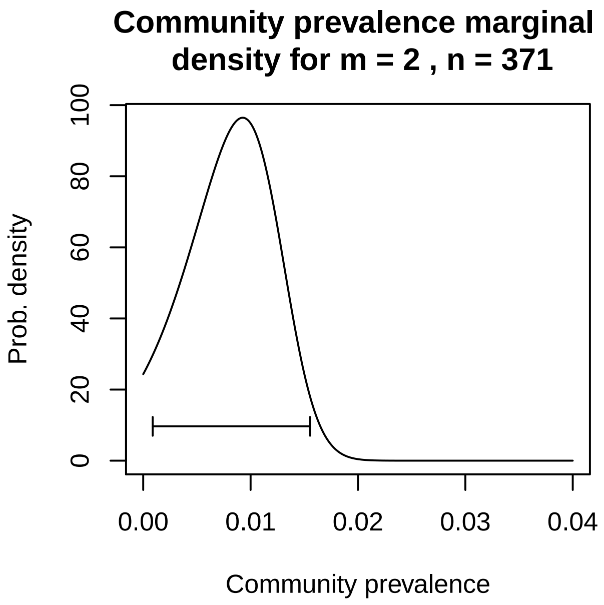
$$\phi_P(P) = \int_0^1 \phi_{P,\rho}(P,\rho)d\rho$$

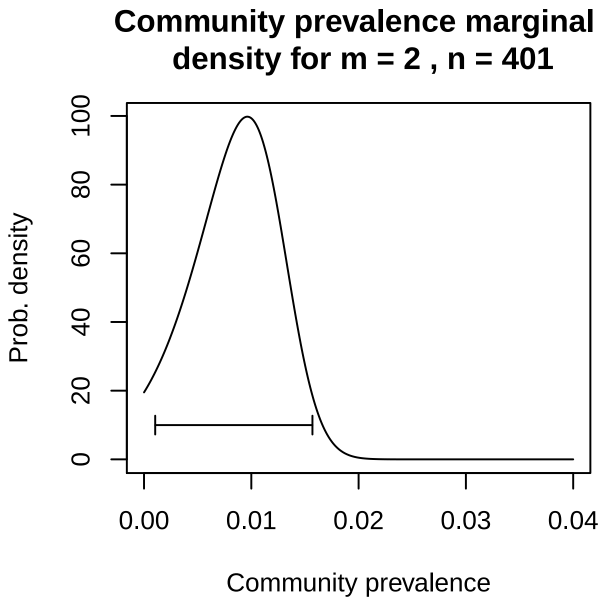
As I mentioned above, I won't mess with trying to find analytic expressions for the normalization or the integrals, but will just do them numerically.

Here are plots of the posterior marginal pdfs for P, with the error bars giving the 95% CI

```r
par(mfrow=c(2,2))

plotPmarg = function(m, n, M, N, nP=1000, nrho=999, Plim=c(0, 0.04), rholim=c(0, 0.04)) {
  rho = seq(rholim[1], rholim[2], length=nrho)
  P = seq(Plim[1], Plim[2], length=nP)
  j = outer(P, rho, jointlikelihood, m=m, n=n, M=M, N=N)
  j = j / sum(j * mean(diff(P)) * mean(diff(rho)))

  Pmarg = apply(j, 1, sum)
  Pmarg = Pmarg / sum(Pmarg*mean(diff(P)))
  plot(P, Pmarg, typ='l', xlab="Community prevalence", ylab="Prob. density",
       main=paste("Community prevalence marginal \n density for m =", m, ", n =", n))

  cPmarg = cumsum(Pmarg*mean(diff(P)))
  margCI = c(P[max(which(cPmarg < 0.025))], P[min(which(cPmarg > 0.975))])
  arrows(margCI[1], 0.1*max(Pmarg), margCI[2], 0.1*max(Pmarg), angle=90, code=3,
         length=0.05)

  mode = P[which.max(Pmarg)]
  mean = sum(P*Pmarg*mean(diff(P)))

  return(list(mean=mean, mode=mode, CI=margCI))

}

m = 0; n = 30
P1 = plotPmarg(m, n, M, N)
m = 2; n = 371
P2 = plotPmarg(m, n, M, N)
m = 2; n = 401
P3 = plotPmarg(m, n, M, N)
```







The 95% CI for prevalence P all go very close to zero. Here is a table of the point estimates and the 95% CIs:

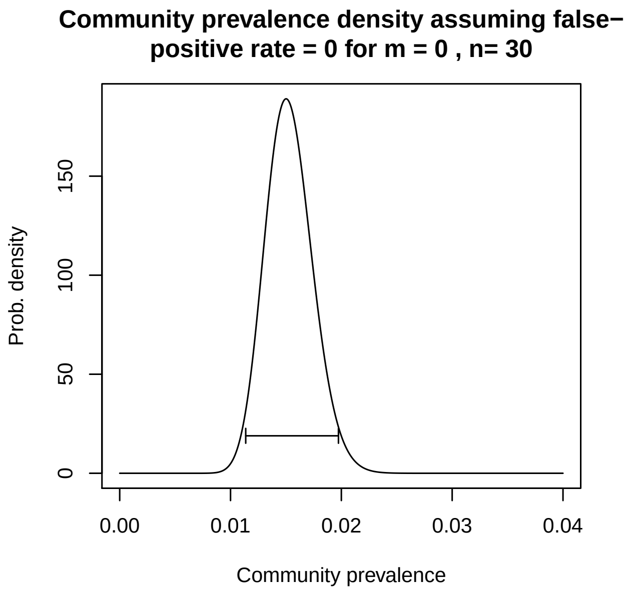
	mean	mode	95% CI
$m = 0, n = 30$	0.0084	0.011	$4.4 \times 10^{-4}, 0.017$
$m = 2, n = 371$	0.0083	0.0092	$8.8 \times 10^{-4}, 0.016$
$m = 2, n = 401$	0.0087	0.0096	$0.001, 0.016$

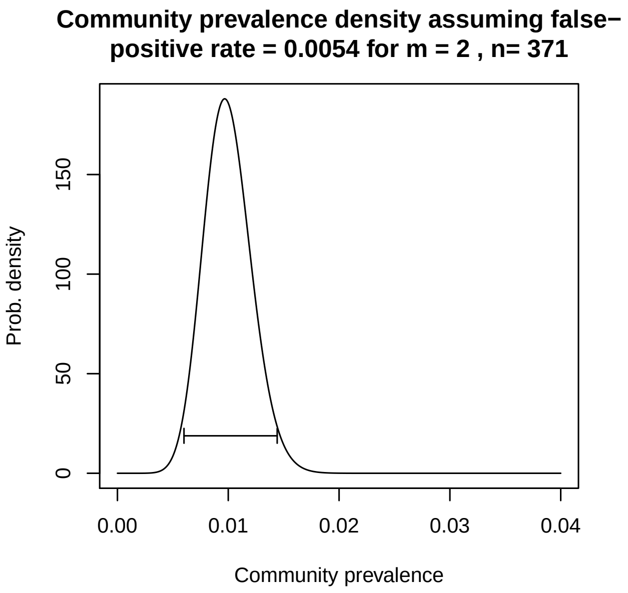
They don't give numbers to compare directly to these. In their Table 2 these numbers would be labeled "Test-performace adjusted (only)" but they only have "Unadjusted", "Population adjusted (only)", and "Population + test-performance adjusted". There is not enough information to reproduce their population adjustment, but this adjustment roughly multiplied their estimated prevalences (point estimates and CIs) by a factor of about 2. But the lower ends of their 95% CIs are factors of 41, 29, 19, respectively, higher than those of the correct test-performance but not-population-adjusted CIs above.

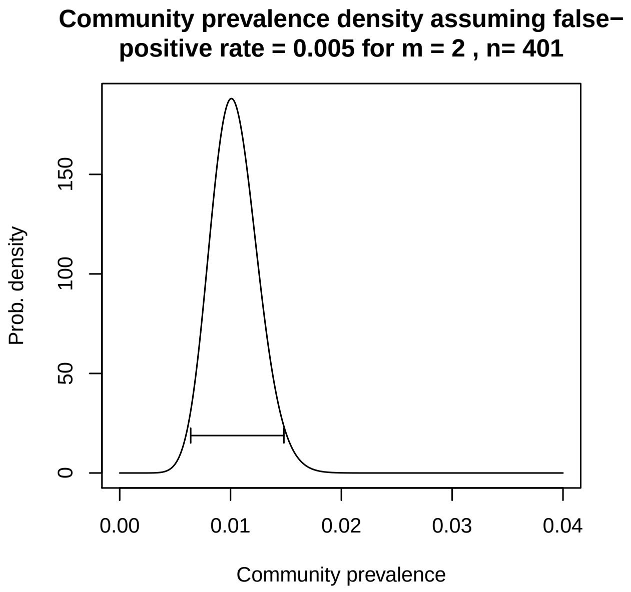
From slices of the joint pdf (the wrong way)

What I think they probably did is to not take the distribution of possible false-positive rates ρ into account and instead just used point estimates of ρ. This is equivalent to using as a distribution a slice of the joint pdf (at the ρ point estimate) instead of the proper marginal distribution. Let's see if that gives results more like what they get:

```
par(mfrow=c(2,2))

plotPslice = function(m, n, M, N, nP=1000, nrho=999, Plim=c(0, 0.04), rholim=c(0, 0.04)) {
  rho = seq(rholim[1], rholim[2], length=nrho)
  P = seq(Plim[1], Plim[2], length=nP)
  j = outer(P, rho, jointlikelihood, m=m, n=n, M=M, N=N)
  j = j / sum(j * mean(diff(P)) * mean(diff(rho)))

  rhohat = m/n

  Pslice = jointlikelihood(P, rhohat, m=m, n=n, M=M, N=N)
  Pslice = Pslice / sum(Pslice*mean(diff(P)))
  plot(P, Pslice, typ='l', xlab="Community prevalence", ylab="Prob. density",
       main=paste("Community prevalence density assuming false-\npositive rate =",
                  signif(m/n, digits=2), "for m =", m, ", n=", n))

  cPslice = cumsum(Pslice*mean(diff(P)))
  sliceCI = c(P[max(which(cPslice < 0.025))], P[min(which(cPslice > 0.975))])
  arrows(sliceCI[1], 0.1*max(Pslice), sliceCI[2], 0.1*max(Pslice), angle=90, code=3,
         length=0.05)

  mode = P[which.max(Pslice)]
  mean = sum(P*Pslice*mean(diff(P)))

  return(list(mean=mean, mode=mode, CI=sliceCI))

}

m = 0; n = 30
P1slice = plotPslice(m, n, M, N)
m = 2; n = 371
P2slice = plotPslice(m, n, M, N)
m = 2; n = 401
P3slice = plotPslice(m, n, M, N)
```







A table of the resulting point estimates and 95% CIs:

	mean	mode	95% CI
$m = 0, n = 30$	0.015	0.015	$0.011, 0.02$
$m = 2, n = 371$	0.01	0.0097	$0.006, 0.014$
$m = 2, n = 401$	0.01	0.01	$0.0064, 0.015$

Now the lower ends of the preprint's prevalence 95% CIs are factors of 1.6, 4.3, 3.1, respectively, higher than those of these *incorrect* test-performance but not-population-adjusted CIs, more in line with the approximate factor of 2 you would expect from their population adjustment - indicating that something like this is what they did in the preprint.

Conclusion

It is clear that they did *not* properly take into account the effects of the plausible range of false-positive rates / specificities implied by their test characteristic data and instead used only point estimates of the false-positive rate / specificity. This resulted in the lower ends of their 95% CIs for prevalence being too high by factors of 10-20.

Taking non-perfect sensitivity into account

If knew sensitivity exactly

If we knew the false-negative rate exactly, call it η, then the probability of getting a positive result in community test will be $P(1-\eta)+\rho(1-P) = P(1-\eta-\rho)+\rho$ and the probability of a negative test $1-P(1-\eta-\rho)-\rho$, so the joint likelihood (and therefore the posterior pdf assuming uniform priors) for ρ and P will be modified to

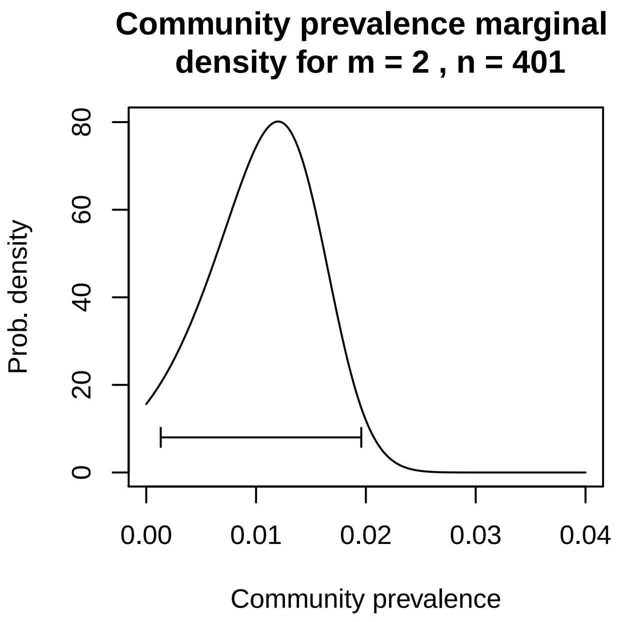
$$\phi_{P,\rho} \propto \rho^m(1-\rho)^{n-m}\left(P(1-\eta-\rho)+\rho\right)^M\left(1-P(1-\eta-\rho)-\rho\right)^{N-M}$$

Let's first just use this with their combined point estimate of sensitivity of 80.3% (so $\eta = 0.197$) to see how this changes the estimates of prevalence.

```r
# note: unnormalized
jointlikelihoodExactEta = function(P, rho, eta, m, n, M, N) {
    rho^m * (1-rho)^(n-m) * (P*(1-eta-rho) + rho)^M * (1 - P*(1-eta-rho) - rho)^(N-M)
}
```

```r
par(mfrow=c(2,2))

plotPmargExactEta = function(m, n, M, N, eta=0.197, nP=1000, nrho=999, Plim=c(0, 0.04), rholim=c(0, 0.04
  rho = seq(rholim[1], rholim[2], length=nrho)
  P = seq(Plim[1], Plim[2], length=nP)
  j = outer(P, rho, jointlikelihoodExactEta, m=m, n=n, M=M, N=N, eta=eta)
  j = j / sum(j * mean(diff(P)) * mean(diff(rho)))

  Pmarg = apply(j, 1, sum)
  Pmarg = Pmarg / sum(Pmarg*mean(diff(P)))
  plot(P, Pmarg, typ='l', xlab="Community prevalence", ylab="Prob. density",
       main=paste("Community prevalence marginal \n density for m =", m, ", n =", n))

  cPmarg = cumsum(Pmarg*mean(diff(P)))
  margCI = c(P[max(which(cPmarg < 0.025))], P[min(which(cPmarg > 0.975))])
  arrows(margCI[1], 0.1*max(Pmarg), margCI[2], 0.1*max(Pmarg), angle=90, code=3,
         length=0.05)

  mode = P[which.max(Pmarg)]
  mean = sum(P*Pmarg*mean(diff(P)))

  return(list(mean=mean, mode=mode, CI=margCI))

}
```

```r
eta = 0.197
P3ExactEta = plotPmargExactEta(m, n, M, N, eta)
```



I'll just use the combined results of their two specificity tests (so 399 negatives out of 401 test) from here out. The point estimates and 95% CIs are a bit higher assuming an η of 0.197 (instead previously assuming it was zero). In the table below, I repeat the $\eta = 0$ results from before along with those for $\eta = 0.197$. The 95% CIs still all go very close to zero.

	mean	mode	95% CI
$m = 2, n = 401, \eta = 0$	0.0087	0.0096	$0.001, 0.016$
$m = 2, n = 401, \eta = 0.197$	0.011	0.012	$0.0013, 0.02$

Including uncertainties in estimates of sensitivity

To test the sensitivity / false-negative rate, the test is applied to known positive samples. They report two sets of numbers for this as well (and, actually, a third but don't seem to use those). In one, testing 37 known positive samples gave 25 positive results; in the other testing 85 known positive samples gave 78 positive results. For a true false-negative rate of η the probability of getting a positive result is $1 - \eta$ and the probability of a positive results is η. So the joint likelihood (again, and therefore the posterior pdf assuming uniform priors) for ρ, P, and η given that there were m positive results out of n tests of known negative samples, μ positive results out of ν test of known positive samples and M positive results out of N population samples is

$$\phi_{P,\rho,\eta} \propto \rho^m(1-\rho)^{n-m}(1-\eta)^{\mu}\eta^{\nu-\mu}\left(P(1-\eta-\rho)+\rho\right)^M\left(1-P(1-\eta-\rho)-\rho\right)^{N-M}$$

```r
# note: unnormalized
jointlikelihoodEta = function(P, rho, eta, m, n, mu, nu, M, N) {
    rho^m * (1-rho)^(n-m) * (1-eta)^mu * eta^(nu-mu) * (P*(1-eta-rho) + rho)^M * (1 - P*(1-eta-rho) - rh
}
```

Here it will be more difficult to plot the full joint likelihood since it is a function of three variables instead of two. Can plot the three 2-d marginal pdfs of the the variables taken two at a time with the other marginalized out. And, of course, the 1-d marginal pdfs with both other variables marginalized out. With the added dimension, my previous evenly sampling the pdf and then approximating intergrals with Riemann sums is too inefficient, so need to do something better. Tried `R`'s builtin 1-d adaptive quadrature routine `integrate()`, but it didn't succeed - claimed the intergrals were probably divergent. The `cubature` packages `hcubature()` seems to be robust but is pretty slow on for these integrals. Are 3 dimensions enough to make it worth going to MCMC? Or perhaps look into how to use `cubature`'s vector interface, which the documentation says can speed things up immensely.

I'll first define the unnormalized pdfs, then find the normalization constant separately.

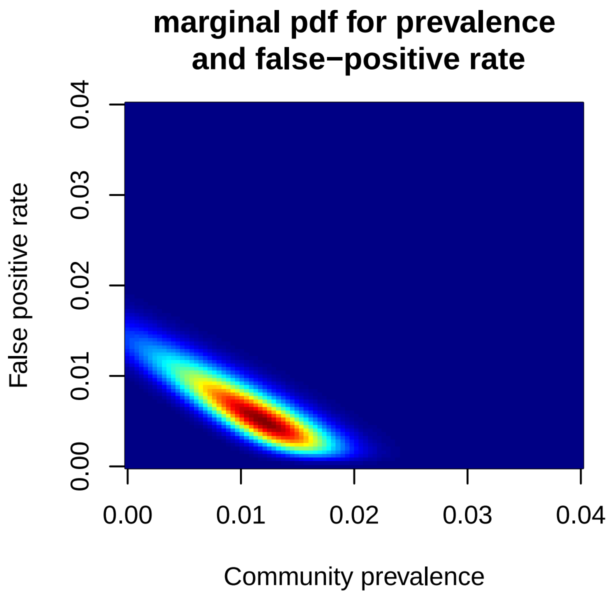
```r
# I'll use the naming convention likelihood_P.rho.eta() is the full 3-d joint
# likelihood, and will remove variable names after the "_" as they are marginalized out.
# Again these are all unnormalized
likelihood_P.rho.eta = jointlikelihoodEta

likelihood_P.rho = function(P, rho, m, n, mu, nu, M, N) {
  stopifnot(length(P) == length(rho) || length(rho) == 1 || length(P) == 1)

  if (length(rho) == 1) rho = rep(rho, length(P))
  if (length(P) == 1) P = rep(P, length(rho))

  # hcubature(f, ...) needs the first arg of f() to the one it integrates over
  f = function(eta, P, rho, m, n, mu, nu, M, N) {likelihood_P.rho.eta(P, rho, eta, m, n, mu, nu, M, N)}

  int = rep(NA, length(P))
  for (ii in 1:length(P))
    int[ii] = hcubature(f, 0, 1, P=P[ii], rho=rho[ii], m=m, n=n, mu=mu, nu=nu, M=M, N=N)$integral

  return(int)
}

likelihood_P.eta = function(P, eta, m, n, mu, nu, M, N) {
  stopifnot(length(P) == length(eta) || length(eta) == 1 || length(P) == 1)

  if (length(eta) == 1) eta = rep(eta, length(P))
  if (length(P) == 1) P = rep(P, length(eta))

  # hcubature(f, ...) needs the first arg of f() to the one it integrates over
  f = function(rho, P, eta, m, n, mu, nu, M, N) {likelihood_P.rho.eta(P, rho, eta, m, n, mu, nu, M, N)}

  int = rep(NA, length(P))
  for (ii in 1:length(P))
    int[ii] = hcubature(f, 0, 1, P=P[ii], eta=eta[ii], m=m, n=n, mu=mu, nu=nu, M=M, N=N)$integral

  return(int)
}

likelihood_rho.eta = function(rho, eta, m, n, mu, nu, M, N) {
  stopifnot(length(rho) == length(eta) || length(rho) == 1 || length(eta) == 1)

  if (length(rho) == 1) rho = rep(rho, length(eta))
  if (length(eta) == 1) eta = rep(eta, length(rho))

  # integrate(f, ...) needs the first arg of f() to the one it integrates over
  f = function(P, rho, eta, m, n, mu, nu, M, N) {likelihood_P.rho.eta(P, rho, eta, m, n, mu, nu, M, N)}
  int = rep(NA, length(rho))
  for (ii in 1:length(rho))
    int[ii] = hcubature(f, 0, 1, rho=rho[ii], eta=eta[ii], m=m, n=n, mu=mu, nu=nu, M=M, N=N)$integral

  return(int)
}

likelihood_P = function(P, m, n, mu, nu, M, N) {
  f = function(rho.eta, P, m, n, mu, nu, M, N) {likelihood_P.rho.eta(P, rho.eta[1], rho.eta[2], m, n, mu

  int = rep(NA, length(P))
  for (ii in 1:length(P))
    int[ii] = hcubature(f, c(0,0), c(1,1), P=P[ii], m=m, n=n, mu=mu, nu=nu, M=M, N=N)$integral

  return(int)
}

likelihood_rho = function(rho, m, n, mu, nu, M, N) {
  f = function(P.eta, rho, m, n, mu, nu, M, N) {likelihood_P.rho.eta(P.eta[1], rho, P.eta[2], m, n, mu,

  int = rep(NA, length(rho))
  for (ii in 1:length(rho))
    int[ii] = hcubature(f, c(0,0), c(1,1), rho=rho[ii], m=m, n=n, mu=mu, nu=nu, M=M, N=N)$integral

  return(int)
}

likelihood_eta = function(eta, m, n, mu, nu, M, N) {
  f = function(P.rho, eta, m, n, mu, nu, M, N) {likelihood_P.rho.eta(P.rho[1], P.rho[2], eta, m, n, mu,

  int = rep(NA, length(eta))
  for (ii in 1:length(eta))
    int[ii] = hcubature(f, c(0,0), c(1,1), eta=eta[ii], m=m, n=n, mu=mu, nu=nu, M=M, N=N)$integral

  return(int)
}

# normalization constant
K = function(m, n, mu, nu, M, N) {
  f = function(P.rho.eta, m, n, mu, nu, M, N) {
    likelihood_P.rho.eta(P.rho.eta[1], P.rho.eta[2], P.rho.eta[3], m, n, mu, nu, M, N)
  }

  hcubature(f, c(0,0,0), c(1,1,1), m=m, n=n, mu=mu, nu=nu, M=M, N=N)$integral
}
```

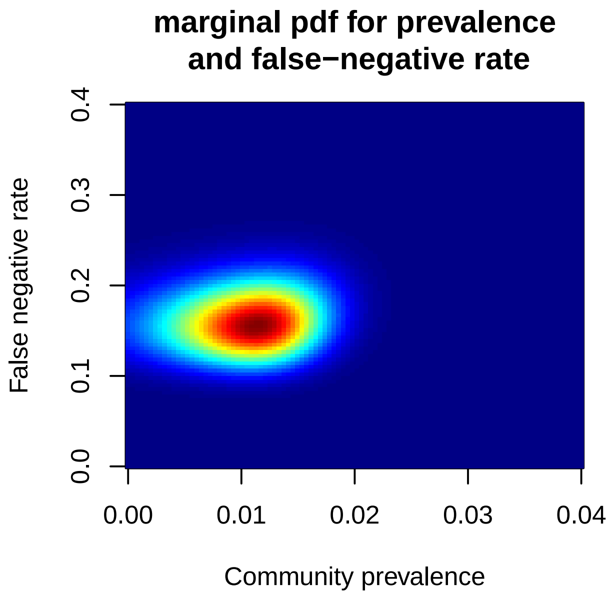
2-D joint marginal posterior pdf plots

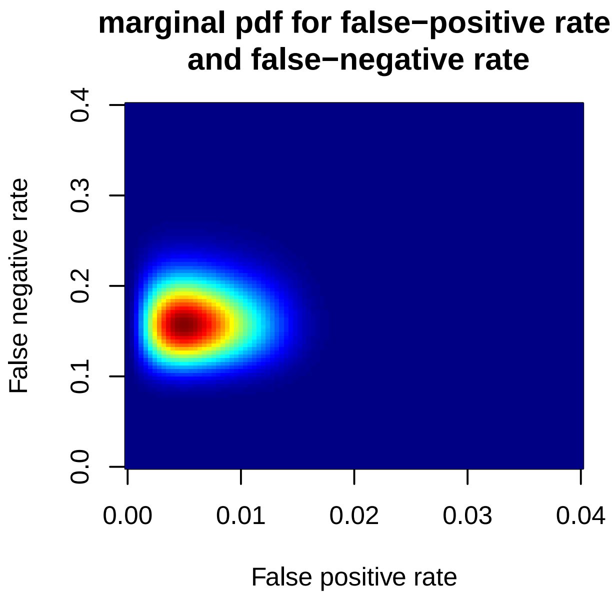
Note the very different scale for false-negative rate compared to false-positive rate and prevalence.

```
m = 2; n = 401; mu = 103; nu = 122; M = 50; N = 3330
K1 = K(m, n, mu, nu, M, N)
```

```r
Plim = c(0, 0.04); nP = 100
rholim = c(0, 0.04); nrho = 101
etalim = c(0, 0.4); neta = 99
P = seq(Plim[1], Plim[2], length=nP)
rho = seq(rholim[1], rholim[2], length=nrho)
eta = seq(etalim[1], etalim[2], length=neta)

par(mfrow=c(2,2))

j = outer(P, rho, likelihood_P.rho, m=m, n=n, mu=mu, nu=nu, M=M, N=N)/K1
#j = j / sum(j * mean(diff(P)) * mean(diff(rho)))
image(P, rho, j, col=jet(100), xlab="Community prevalence", ylab="False positive rate",
      main=paste("marginal pdf for prevalence\n and false-positive rate"))

j = outer(P, eta, likelihood_P.eta, m=m, n=n, mu=mu, nu=nu, M=M, N=N)/K1
#j = j / sum(j * mean(diff(P)) * mean(diff(rho)))
image(P, eta, j, col=jet(100), xlab="Community prevalence", ylab="False negative rate",
      main=paste("marginal pdf for prevalence\n and false-negative rate"))

j = outer(rho, eta, likelihood_rho.eta, m=m, n=n, mu=mu, nu=nu, M=M, N=N)/K1
#j = j / sum(j * mean(diff(P)) * mean(diff(rho)))
image(rho, eta, j, col=jet(100), xlab="False positive rate", ylab="False negative rate",
      main=paste("marginal pdf for false-positive rate\n and false-negative rate"))
```





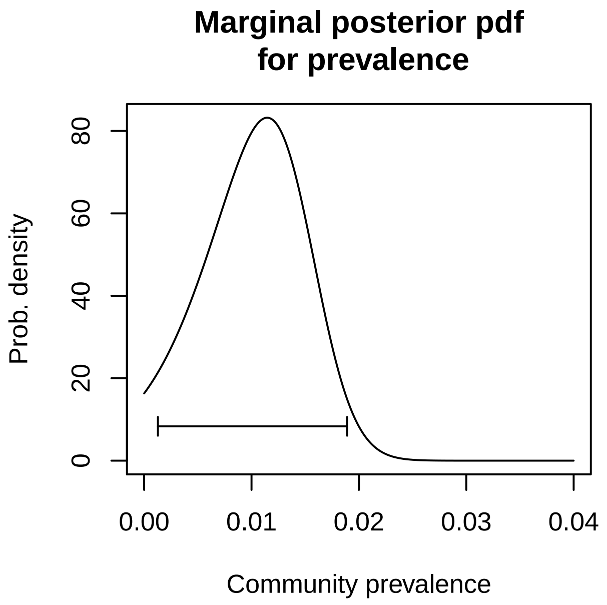


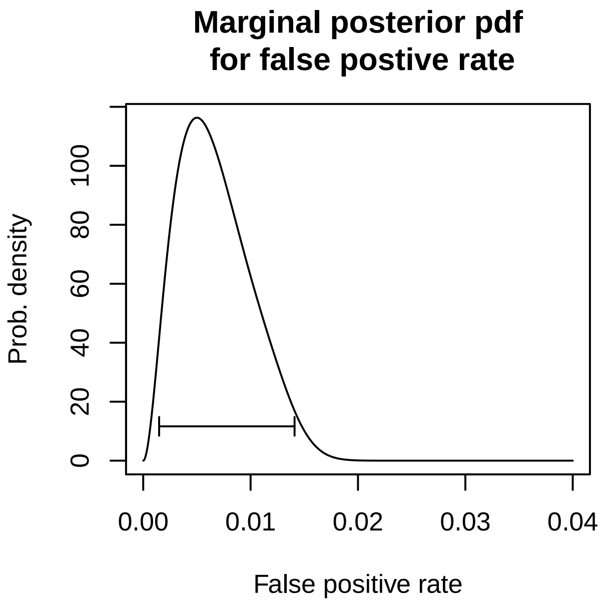
1-D marginal pdf plots

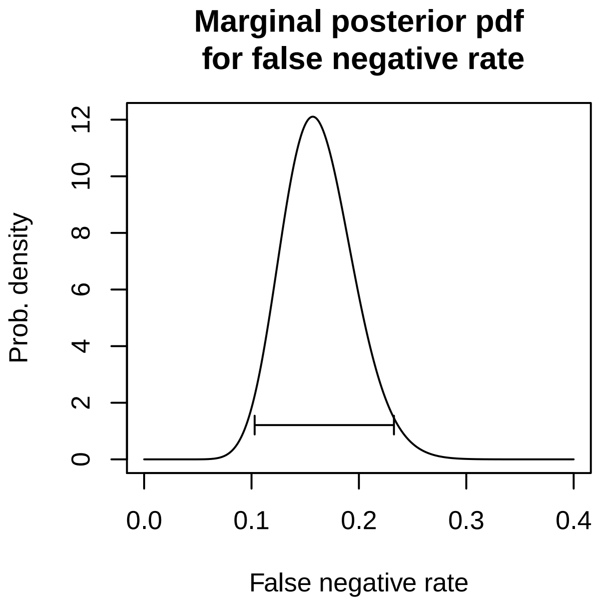
Again, note the very different scale for false negative rate than for the prevalence or the false positive rate.

```
par(mfrow=c(2,2))
Plim = c(0, 0.04); nP = 1000
rholim = c(0, 0.04); nrho = 1000
etalim = c(0, 0.4); neta = 1000
P = seq(Plim[1], Plim[2], length=nP)
rho = seq(rholim[1], rholim[2], length=nrho)
eta = seq(etalim[1], etalim[2], length=neta)

dP = likelihood_P(P, m, n, mu, nu, M, N)/K1
cP = cumsum(dP*mean(diff(P)))
```

```
PCI = c(P[max(which(cP < 0.025))], P[min(which(cP > 0.975))])
Pmode = P[which.max(dP)]
Pmean = sum(P*dP*mean(diff(P)))
plot(P, dP, typ='l',
     xlab="Community prevalence", ylab="Prob. density", main="Marginal posterior pdf\n for prevalence")
arrows(PCI[1], 0.1*max(dP), PCI[2], 0.1*max(dP), angle=90, code=3,
         length=0.05)

drho = likelihood_rho(rho, m, n, mu, nu, M, N)/K1
crho = cumsum(drho*mean(diff(rho)))
rhoCI = c(rho[max(which(crho < 0.025))], rho[min(which(crho > 0.975))])
rhomode = rho[which.max(drho)]
rhomean = sum(rho*drho*mean(diff(rho)))

plot(rho, drho, typ='l',
     xlab="False positive rate", ylab="Prob. density", main="Marginal posterior pdf\n for false postive
arrows(rhoCI[1], 0.1*max(drho), rhoCI[2], 0.1*max(drho), angle=90, code=3,
         length=0.05)


deta = likelihood_eta(eta, m, n, mu, nu, M, N)/K1
ceta = cumsum(deta*mean(diff(eta)))
etaCI = c(eta[max(which(ceta < 0.025))], eta[min(which(ceta > 0.975))])
etamode = eta[which.max(deta)]
etamean = sum(eta*deta*mean(diff(eta)))
plot(eta, deta, typ='l',
     xlab="False negative rate", ylab="Prob. density", main="Marginal posterior pdf\n for false negativ
arrows(etaCI[1], 0.1*max(deta), etaCI[2], 0.1*max(deta), angle=90, code=3,
         length=0.05)
```







A table of posterior point estimates and 95% CIs, with the earlier prevalence results assuming a zero and 0.197 false negative rates on the second and third lines for comparison:

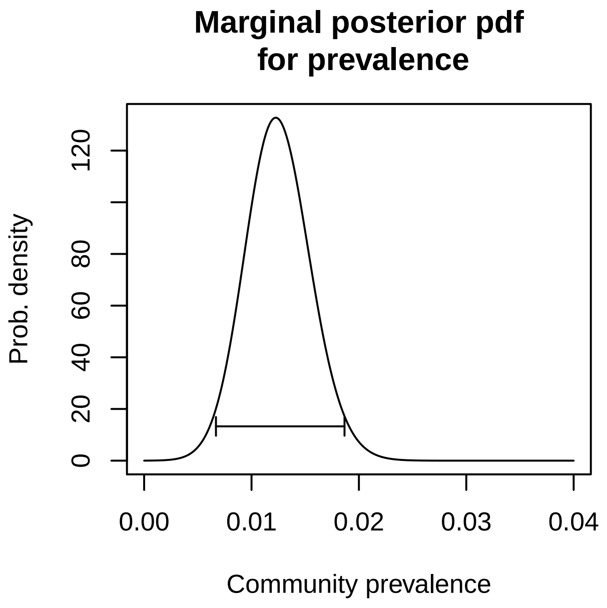
	mean	mode	95% CI
prevalence	0.01	0.011	$0.0013, 0.019$
prevalence assuming $\eta = 0$	0.0087	0.0096	$0.001, 0.016$
prevalence assuming $\eta = 0.197$	0.011	0.012	$0.0013, 0.02$
false positive rate	0.0068	0.005	$0.0015, 0.014$
false negative rate	0.16	0.16	$0.1, 0.23$

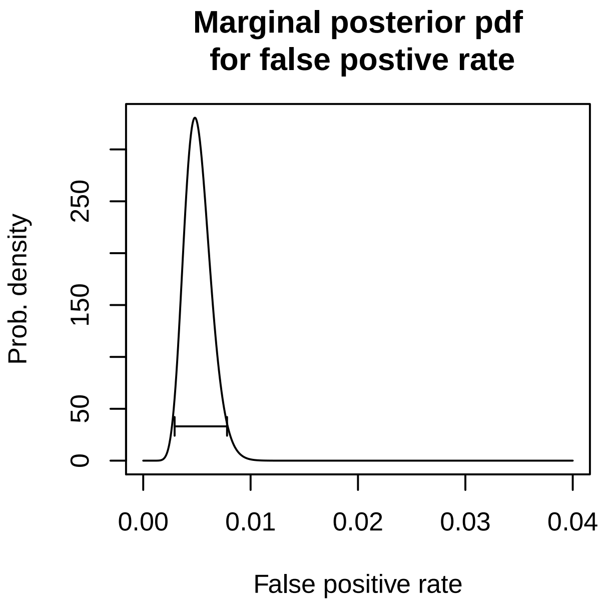
As I suspected, including the effects of a non-zero false negative rate (and uncertainties thereof) did not change the prevalence estimates or 95% CIs all that much - the lower end of the 95% CI still goes very close to zero (0.13%).

Updated study

The same authors put an updated preprint at: https://www.medrxiv.org/content/10.1101/2020.04.14.20062463v2.full.pdf. One change is they have more data on the specificity and sensitivity. Redoing the above analysis with their new numbers:

```
M = 50; N = 3330; m = 16; n = 3324; mu = 130; nu = 157
K1 = K(m, n, mu, nu, M, N)

par(mfrow=c(2,2))

dP.2 = likelihood_P(P, m, n, mu, nu, M, N)/K1
cP.2 = cumsum(dP.2*mean(diff(P)))
PCI.2 = c(P[max(which(cP.2 < 0.025))], P[min(which(cP.2 > 0.975))])
Pmode.2 = P[which.max(dP.2)]
Pmean.2 = sum(P*dP.2*mean(diff(P)))
plot(P, dP.2, typ='l',
     xlab="Community prevalence", ylab="Prob. density", main="Marginal posterior pdf\n for prevalence")
arrows(PCI.2[1], 0.1*max(dP.2), PCI.2[2], 0.1*max(dP.2), angle=90, code=3,
         length=0.05)

drho.2 = likelihood_rho(rho, m, n, mu, nu, M, N)/K1
crho.2 = cumsum(drho.2*mean(diff(rho)))
rhoCI.2 = c(rho[max(which(crho.2 < 0.025))], rho[min(which(crho.2 > 0.975))])
rhomode.2 = rho[which.max(drho.2)]
rhomean.2 = sum(rho*drho.2*mean(diff(rho)))

plot(rho, drho.2, typ='l',
     xlab="False positive rate", ylab="Prob. density", main="Marginal posterior pdf\n for false postive
arrows(rhoCI.2[1], 0.1*max(drho.2), rhoCI.2[2], 0.1*max(drho.2), angle=90, code=3,
         length=0.05)


deta.2 = likelihood_eta(eta, m, n, mu, nu, M, N)/K1
ceta.2 = cumsum(deta.2*mean(diff(eta)))
etaCI.2 = c(eta[max(which(ceta.2 < 0.025))], eta[min(which(ceta.2 > 0.975))])
etamode.2 = eta[which.max(deta.2)]
etamean.2 = sum(eta*deta.2*mean(diff(eta)))
plot(eta, deta.2, typ='l',
     xlab="False negative rate", ylab="Prob. density", main="Marginal posterior pdf\n for false negativ
arrows(etaCI.2[1], 0.1*max(deta.2), etaCI.2[2], 0.1*max(deta.2), angle=90, code=3,
         length=0.05)
```





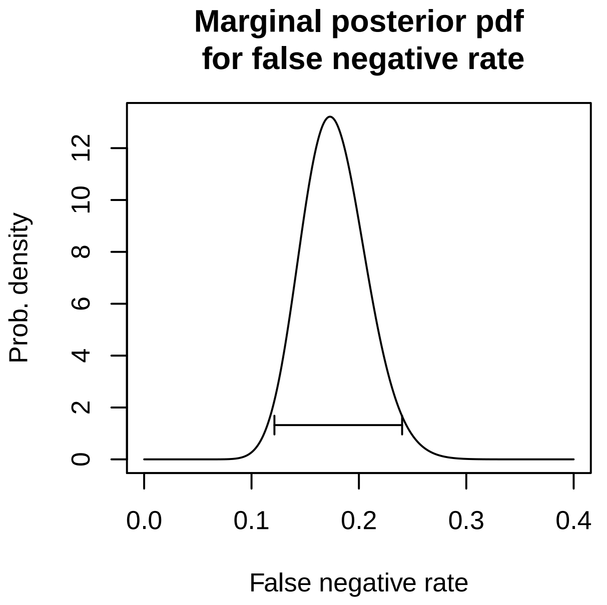


Table repeated from above with added lines for the updated preprint:

	mean	mode	95% CI
prevalence - updated	0.012	0.012	$0.0067, 0.019$
prevalence	0.01	0.011	$0.0013, 0.019$
prevalence assuming $\eta = 0$	0.0087	0.0096	$0.001, 0.016$
prevalence assuming $\eta = 0.197$	0.011	0.012	$0.0013, 0.02$
false positive rate - updated	0.0051	0.0048	$0.0029, 0.0078$
false positive rate	0.0068	0.005	$0.0015, 0.014$
false negative rate - updated	0.18	0.17	$0.12, 0.24$
false negative rate	0.16	0.16	$0.1, 0.23$

The extra data on the specificity has cut the upper end of the 95% CI on the false positive rate in half, and thus the lower end of the 95% CI on the prevalence has increased from 0.13% to 0.67%. In their updated preprint they do now give an estimate and 95% CI for the prevalence including the test-adjustments but not population adjustments, so we can directly compare our results here. Gratifyingly, they now give a point estimate of 1.2% and a 95% CI of 0.7% to 1.8%, compared to our 1.2% and 0.7% to 1.9%. I haven't read the updated preprint in detail to see if this agreement is because they now properly take into account the uncertainties in the specificity and sensitivity or because now the specificity is tightly constrained enough by the data that it doesn't matter.