Benchmarking Moments++

Gustavo V. Barroso

2023-11-03

Neutrality

No-migration

Running moments++

Running moments.LD

```python
import moments.LD
import demes
import sys, os
import numpy as np

def simulate_ld(f, sampled_demes):
    g = demes.load(f)
    u = 1e-7
    r = 1e-6

    y = moments.LD.LDstats.from_demes(g, sampled_demes, r=r, u=u)
    return y

def write_column(y, out_file):
    mld, mh = moments.LD.Util.moment_names(y.num_pops)
    # write D2
    for m, v in zip(mld, y[0]):
        if m.split("_")[0] == "DD":
            out_file.write(str(v) + "\n")
    # write Dz
    for m, v in zip(mld, y[0]):
        if m.split("_")[0] == "Dz":
            out_file.write(str(v) + "\n")
    # write H
    for m, v in zip(mh, y[-1]):
        out_file.write(str(v) + "\n")
    # write pi2
    for m, v in zip(mld, y[0]):
        if m.split("_")[0] == "pi2":
            out_file.write(str(v) + "\n")
```

```
for i in range(1, 5):
  with open(f"neutrality/no_migration/moments_LD_model_{i}.csv", "w+") as fout:
      f = f"neutrality/no_migration/model_{i}_demes.yaml"
      pops = ["A", "B"]
      y = simulate_ld(f, pops)
      stats = moments.LD.Util.moment_names(len(pops))
      l = ",".join(stats[0] + stats[1]) + "\n"
      fout.write(l)
      l = ",".join([str(_) for _ in y[0]] + [str(_) for _ in y[1]]) + "\n"
      fout.write(l)
```

Plots

```
n_models <- 4

# considering symmetries under neutrality
core_stats <- c("DD_1_1",
                "DD_1_2",
                "DD_2_2",
                "Dr_1_1_(1-2p1)^1",
                "Dr_1_1_(1-2p2)^1",
                "Dr_1_2_(1-2p2)^1",
                "Dr_2_1_(1-2p1)^1",
                "Dr_2_1_(1-2p2)^1",
                "Dr_2_2_(1-2p2)^1",
                "pi2_1_1_1_1",
                "pi2_1_1_1_2",
                "pi2_1_1_2_2",
                "pi2_1_2_1_2",
                "pi2_1_2_2_2",
                "pi2_2_2_2_2",
                "Hl_1_1",
                "Hl_1_2",
                "Hl_2_2")

# constant Ne
mpp <- numeric()
for(i in 1:n_models) {

  x <- read.table(paste("neutrality/no_migration/model_", i, "_expectations.txt", sep=""))
  x <- subset(x, V1 %in% core_stats)
  x <- slice(x, 1:9, 13:18, 10:12) # to match moments.LD output
  mpp <- c(mpp, x$V3)
}

tbl <- as.data.frame(mpp)
tbl$stats <- core_stats
scenario <- NULL
for(i in 1:n_models) {
  scenario <- c(scenario, rep(i, length(core_stats)))
}
tbl$scenario <- scenario
```

```
mLD <- numeric()
for(i in 1:n_models) {

  y <- as.numeric(read.csv(paste("neutrality/no_migration/moments_LD_model_", i, ".csv", sep=""), heade
  mLD <- c(mLD, y)
}

tbl$mLD <- mLD
tbl <- select(tbl, scenario, stats, mpp, mLD)
tbl$ratio <- tbl$mpp / tbl$mLD

molten_tbl <- pivot_longer(tbl, cols=starts_with("m"))

p1 <- ggplot(data=molten_tbl, aes(x=stats, y=ratio, shape=as.factor(scenario)))
p1 <- p1 + geom_point(size=3) + theme_bw()
p1 <- p1 + geom_hline(yintercept=0.99, linetype=2)
p1 <- p1 + geom_hline(yintercept=1.01, linetype=2)
p1 <- p1 + labs(title="++ / LD", x="Moment", y="Ratio", shape="Model")
p1 <- p1 + scale_shape_manual(values=(0:(n_models - 1)))
p1 <- p1 + theme(axis.title=element_text(size=12),
                 axis.text=element_text(size=10),
                 axis.text.x=element_text(angle=90, size=8, vjust=0.5, hjust=1.0),
                 legend.position="bottom")

save_plot("neutrality/no_migration/ratios_no-mig.pdf", p1, base_height=12, base_width=12)
```

Selection

No-migration

Building Demes files

```
# symmetrical pop sizes
N1 <- 10000
N2 <- c(N1, N1 / 10)
u <- 1e-7
r  <- c(1e-3, 1e-4, 1e-5, 1e-6, 0)
s1 <- c(0, -5e-05, -1e-04, -2.5e-04, -5e-04, -1e-3)
s2 <- c(0, -5e-05, -1e-04, -2.5e-04, -5e-04, -1e-3)

split_time <- 2000

params <- crossing(N1, N2, u, r, s1, s2)
params$scenario <- 1:nrow(params)
n_models <- nrow(params)

for(i in 1:n_models) {

  name <- paste("model_", i, sep="")
  sink(paste("selection/no_migration/", name, ".yaml", sep=""))
```

```r
cat("time_units: generations\n")
cat("demes:\n")
cat("  - name: X\n")
cat("    epochs:\n")
cat("      - end_time: ")
cat(split_time)
cat("\n")
cat("        start_size: ")
cat(params$N1[i])
cat("\n")
cat("  - name: A\n")
cat("    ancestors: [X]\n")
cat("    epochs:\n")
cat("      - end_time: 0\n")
cat("        start_size: ")
cat(params$N1[i])
cat("\n")
cat("  - name: B\n")
cat("    ancestors: [X]\n")
cat("    epochs:\n")
cat("      - end_time: 0\n")
cat("        start_size: ")
cat(params$N2[i])
cat("\n")
cat("metadata:\n")
cat("  - name: mutation\n")
cat("    epochs:\n")
cat("      - end_time: 0\n")
cat("        rates: [")
cat(params$u[i])
cat("]\n")
cat("  - name: recombination\n")
cat("    epochs:\n")
cat("      - end_time: 0\n")
cat("        rates: [")
cat(params$r[i])
cat("]\n")
cat("  - name: selection\n")
cat("    start_time: .inf\n")
cat("    epochs:\n")
cat("      - end_time: ")
cat(split_time)
cat("\n")
cat("        rates: [")
cat(-1/2/N1)
cat("]\n")
cat("      - end_time: 0\n")
cat("        rates: [")
cat(params$s1[i])
cat(", ")
cat(params$s2[i])
cat("]\n")
```

```
  sink()
}
```

Running moments++

Plots

```
core_stats <- c("DD_1_1",
                "DD_1_2",
                "DD_2_2",
                "Dr_1_1_(1-2p1)^1",
                "Dr_1_1_(1-2p2)^1",
                "Dr_1_2_(1-2p2)^1",
                "Dr_2_1_(1-2p1)^1",
                "Dr_2_1_(1-2p2)^1",
                "Dr_2_2_(1-2p2)^1",
                "Hl_1_1",
                "Hl_1_2",
                "Hl_2_1",
                "Hl_2_2",
                "Hr_1_1",
                "Hr_1_2",
                #"Hr_2_1",
                "Hr_2_2",
                "pi2_1_1_1_1",
                "pi2_1_1_1_2",
                "pi2_1_1_2_2",
                "pi2_1_2_1_2",
                "pi2_1_2_2_2",
                "pi2_2_2_2_2")

expectation <- numeric()
for(i in 1:n_models) {

  x <- read.table(paste("selection/no_migration/model_", i, "_expectations.txt", sep=""))
  x <- subset(x, V1 %in% core_stats)
  expectation <- c(expectation, x$V3)
}

tbl <- as.data.frame(expectation)
tbl$stats <- core_stats
scenario <- NULL
for(i in 1:n_models) {
  scenario <- c(scenario, rep(i, length(core_stats)))
}
tbl$scenario <- scenario

df <- full_join(tbl, params, by="scenario")

# transforms variables
df$alpha_1 <- 2 * df$s1 * df$N1
df$alpha_2 <- 2 * df$s2 * df$N2
```

```
df <- df[df$r < 1e-3,] # for increased resolution in the plots

# symmetric population sizes
df_sym_N <- df[df$N1==df$N2,]
df_sym_N[df_sym_N$r==0,]$r <- 1e-12 # for log transformation

# D1D2
d1d2 <- subset(df_sym_N, stats %in% "DD_1_2")
d1d2$ratio <- d1d2$expectation / subset(df_sym_N, stats %in% "pi2_1_2_1_2")$expectation

p1 <- ggplot(data=d1d2, aes(x=r, y=ratio)) + facet_grid(+alpha_1~alpha_2)
p1 <- p1 + geom_point(size=3, shape=16) + geom_line() + theme_bw()
p1 <- p1 + scale_x_log10(breaks=c(1e-12, 1e-6, 1e-5, 1e-4, 1e-3))
p1 <- p1 + scale_y_log10(breaks=c(0.01, 0.05, 0.10, 0.2, 0.3))
p1 <- p1 + labs(title="Symmetric Population Sizes", x="r", y="E[D1D2] / E[pi2_1212]")
p1 <- p1 + theme(axis.title=element_text(size=16),
                 axis.text=element_text(size=12),
                 axis.text.x=element_text(angle=90, size=12, vjust=0.5, hjust=1.0),
                 legend.position="bottom")

save_plot("selection/no_migration/Covar_D_no-mig_1.pdf", p1, base_height=12, base_width=12)

d1d2 <- d1d2[d1d2$alpha_1==d1d2$alpha_2,]

p2 <- ggplot(data=d1d2, aes(x=alpha_1, y=ratio, color=as.factor(r)))
p2 <- p2 + geom_point(size=3) + geom_line() + theme_bw()
p2 <- p2 + scale_color_brewer(palette="PuOr")
p2 <- p2 + scale_y_log10(breaks=c(0.01, 0.05, 0.10, 0.2, 0.3))
p2 <- p2 + labs(title="Symmetric Population Sizes and Selection coefficients", x="2Ns", y="E[D1D2] / E[
p2 <- p2 + theme(axis.title=element_text(size=16),
                 axis.text=element_text(size=12),
                 axis.text.x=element_text(angle=90, size=12, vjust=0.5, hjust=1.0),
                 legend.position="bottom")

save_plot("selection/no_migration/Covar_D_no-mig_2.pdf", p2, base_height=12, base_width=12)

# Fst at the Left Locus
Hl_12 <- subset(df_sym_N, stats %in% "Hl_1_2")
Hl_21 <- subset(df_sym_N, stats %in% "Hl_2_1")
Hl_11 <- subset(df_sym_N, stats %in% "Hl_1_1")
Hl_22 <- subset(df_sym_N, stats %in% "Hl_2_2")

pi_between <- (Hl_12$expectation + Hl_21$expectation) / 2
pi_within <- (Hl_11$expectation + Hl_22$expectation) / 2

fst <- Hl_12
fst$expectation <- 1 - pi_within / pi_between

fst <- fst[fst$alpha_1==fst$alpha_2,]

p3 <- ggplot(data=fst, aes(x=alpha_1, y=expectation))
p3 <- p3 + geom_point(size=3) + geom_line() + theme_bw()
p3 <- p3 + scale_color_brewer(palette="PuOr")
```

```r
p3 <- p3 + scale_y_log10(breaks=pretty_breaks())
p3 <- p3 + labs(title="Symmetric Population Sizes and Selection coefficients", x="2Ns", y="Fst selected
p3 <- p3 + theme(axis.title=element_text(size=16),
                 axis.text=element_text(size=12),
                 axis.text.x=element_text(angle=90, size=12, vjust=0.5, hjust=1.0),
                 legend.position="bottom")

save_plot("selection/no_migration/Fst_no-mig_left.pdf", p3, base_height=12, base_width=12)

# Fst at the Right Locus
Hr_12 <- subset(df_sym_N, stats %in% "Hr_1_2")
#Hr_21 <- subset(df_sym_N, stats %in% "Hr_2_1")
Hr_11 <- subset(df_sym_N, stats %in% "Hr_1_1")
Hr_22 <- subset(df_sym_N, stats %in% "Hr_2_2")

pi_between <- Hr_12$expectation #(Hr_12$expectation + Hr_21$expectation) / 2
pi_within <- (Hr_11$expectation + Hr_22$expectation) / 2

fst <- Hr_12
fst$expectation <- 1 - pi_within / pi_between

p5 <- ggplot(data=fst, aes(x=r, y=expectation)) + facet_grid(+alpha_1~alpha_2)
p5 <- p5 + geom_point(size=3, shape=16) + geom_line() + theme_bw()
p5 <- p5 + scale_x_log10(breaks=c(1e-12, 1e-6, 1e-5, 1e-4, 1e-3))
p5 <- p5 + scale_y_log10(breaks=pretty_breaks())
p5 <- p5 + labs(title="Symmetric Population Sizes", x="r", y="Fst neutral locus")
p5 <- p5 + theme(axis.title=element_text(size=16),
                 axis.text=element_text(size=12),
                 axis.text.x=element_text(angle=90, size=12, vjust=0.5, hjust=1.0),
                 legend.position="bottom")

save_plot("selection/no_migration/Fst_no-mig_right_2.pdf", p5, base_height=12, base_width=12)

fst <- fst[fst$alpha_1==fst$alpha_2,]

p4 <- ggplot(data=fst, aes(x=alpha_1, y=expectation, color=as.factor(r)))
p4 <- p4 + geom_point(size=3) + geom_line() + theme_bw()
p4 <- p4 + scale_color_brewer(palette="PuOr")
p4 <- p4 + scale_y_log10(breaks=pretty_breaks())
p4 <- p4 + labs(title="Symmetric Population Sizes and Selection coefficients", x="2Ns", y="Fst neutral l
p4 <- p4 + theme(axis.title=element_text(size=16),
                 axis.text=element_text(size=12),
                 axis.text.x=element_text(angle=90, size=12, vjust=0.5, hjust=1.0),
                 legend.position="bottom")

save_plot("selection/no_migration/Fst_no-mig_right_1.pdf", p4, base_height=12, base_width=12)
```

Check-my-matrix

```python
import moments.LD
import demes
```

```python
import csv
import numpy as np

np.set_printoptions(threshold=10000)

def pulse_migration(num_pops, pop0, pop1, f):
    """
    Pulse migration from pop0 into pop1 with proportion f.
    """
    # create the matrix that would build a new population via admixture
    A = moments.LD.Matrices.admix_ld(num_pops, pop0, pop1, f)
    # the last population created replaces pop1
    mom_from = moments.LD.Util.moment_names(num_pops + 1)[0]
    mom_to = moments.LD.Util.moment_names(num_pops)[0]
    P = np.zeros((len(mom_to), len(mom_from)))
    for i, m in enumerate(mom_to):
        l = m.split("_")
        for k in range(1, len(l)):
            if l[k] == str(pop1):
                l[k] = str(num_pops)
        m_from = "_".join(l)
        m_from = moments.LD.Util.map_moment(m_from)
        j = mom_from.index(m_from)
        P[i, j] = 1
    return P.dot(A)

# some hard coding never hurt anyone...
mat_LD = pulse_migration(3, 0, 1, 0.3)
matpp = np.loadtxt('multi_epoch/model_2_e_6_admix.csv', delimiter=",")

print(((matpp - mat_LD)).sum())
print(((matpp - mat_LD)**2).sum())

mat_LD = pulse_migration(4, 0, 3, 0.2)
matpp = np.loadtxt('multi_epoch/model_3_e_5_admix.csv', delimiter=",")
print(((matpp - mat_LD)).sum())
print(((matpp - mat_LD)**2).sum())

for i in range(len(mat_LD)):
    if np.abs(mat_LD[i] - matpp[i]).sum() > 1e-15:
        print(i, moments.LD.Util.moment_names(4)[0][i], np.abs(mat_LD[i] - matpp[i]).sum())
        print(mat_LD[i])
        print(matpp[i])

mat_LD = pulse_migration(5, 2, 4, 0.5)
matpp = np.loadtxt('multi_epoch/model_15_e_9_admix.csv', delimiter=",")
print(((matpp - mat_LD)).sum())
print(((matpp - mat_LD)**2).sum())

for i in range(len(mat_LD)):
    if np.abs(mat_LD[i] - matpp[i]).sum() > 1e-15:
        print(i, moments.LD.Util.moment_names(5)[0][i], np.abs(mat_LD[i] - matpp[i]).sum())
        print(mat_LD[i])
```

```
        print(matpp[i])

mig_mat = [
        [0, 1e-4, 0],
        [0, 0, 0],
        [0, 0, 0]
        ]

matpp = np.loadtxt('multi_epoch/model_2_e_5_mig.csv', delimiter=",")
mat_LD = moments.LD.Matrices.migration_ld(3, 2 * mig_mat).todense()
#print(matpp - mat_LD)
print(((matpp - mat_LD)).sum())
print(((matpp - mat_LD)**2).sum())

for i in range(len(mat_LD)):
    if np.abs(mat_LD[i] - matpp[i]).sum() > 1e-15:
        print(i, moments.LD.Util.moment_names(5)[0][i], np.abs(mat_LD[i] - matpp[i]).sum())
        print(mat_LD[i])
        print(matpp[i])

mig_mat = [
        [0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0],
        [0, 0, 0, 0, 0],
        [0, 0, 1e-4, 0, 0]
        ]

matpp = np.loadtxt('multi_epoch/model_15_e_10_mig.csv', delimiter=",")
mat_LD = moments.LD.Matrices.migration_ld(5, 2 * mig_mat).todense()
#print(matpp - mat_LD)
print(((matpp - mat_LD)).sum())
print(((matpp - mat_LD)**2).sum())

for i in range(len(mat_LD)):
    if np.abs(mat_LD[i] - matpp[i]).sum() > 1e-15:
        print(i, moments.LD.Util.moment_names(5)[0][i], np.abs(mat_LD[i] - matpp[i]).sum())
        print(mat_LD[i])
        print(matpp[i])
```