Table S1. Location and information for sequenced populations (DrosRTEC).

State	City	Year	Month	Latitude	Longitude	Altitude (m)	Collector
Florida	Homestead	2008	7	25.47	-80.47	1	Schmidt Lab
Georgia	Hahira	2008	7	30.99	-83.37	69	Schmidt Lab
South Carolina	Eutawville	2008	7	33.40	-80.35	34	Schmidt Lab
North Carolina	Raleigh	2003	NA	35.78	-78.64	106	Trudy Mackay (DGRP)
Virginia	Charlottesville	2012	7	38.03	-78.48	140	Schmidt Lab
Pennsylvania	Media	2012	7	39.53	-75.24	35	Schmidt Lab
New York	Ithaca	2012	9	42.44	-76.50	121	Lazarro Lab
Massachusetts	Lancaster	2012	7	42.46	-71.67	91	Schmidt Lab
Wisconsin	Cross Plains	2012	6	43.09	-89.66	295	Pool Lab
Maine	Bowdoinham	2010	10	44.02	-69.88	17	Schmidt Lab

Table S2. Allelic and line information for construction of the recombinant outbred (ROP) populations. HL denotes the *foxo* genotype common at high latitudes, as defined by the two-SNP haplotype, and LL the foxo genotype at higher incidence at low latitude. The experimental design included two biological replicates, using independent, non-overlapping sets of inbred lines, for each assayed genotype; these replicates are given as Set (A,B for the HL genotype and C,D for the low latitude genotype). Each set was replicated twice, as denoted by Cage. The last column denotes the line numbers for each inbred line from the Drosophila Genetics Reference Panel (DGRP) used to establish each of the eight individual cages.

foxo Genotype	Nucleotide Position	Set	Cage	Lines used (N = 18 per cage)
LL	3R:9892517 / 9894559 (GG)	A	1	26, 57, 73, 75, 91, 101, 105, 161, 176, 280, 313, 318, 367, 371, 375, 377, 378, 379
LL	3R:9892517 / 9894559 (GG)	A	2	26, 57, 73, 75, 91, 101, 105, 161, 176, 280, 313, 318, 367, 371, 375, 377, 378, 379
LL	3R:9892517 / 9894559 (GG)	B	3	208, 373, 406, 426, 440, 491, 492, 508, 513, 535, 639, 646, 757, 761, 796, 805, 812, 852
LL	3R:9892517 / 9894559 (GG)	B	4	208, 373, 406, 426, 440, 491, 492, 508, 513, 535, 639, 646, 757, 761, 796, 805, 812, 852
HL	3R:9892517 / 9894559 (AT)	C	5	40, 41, 42, 69, 83, 109, 142, 153, 158, 177, 195, 229, 233, 365, 370, 380, 391, 405
HL	3R:9892517 / 9894559 (AT)	C	6	40, 41, 42, 69, 83, 109, 142, 153, 158, 177, 195, 229, 233, 365, 370, 380, 391, 405
HL	3R:9892517 / 9894559 (AT)	D	7	45, 332, 338, 443, 517, 531, 595, 703, 705, 707, 774, 790, 804, 820, 837, 855, 879, 890
HL	3R:9892517 / 9894559 (AT)	D	8	45, 332, 338, 443, 517, 531, 595, 703, 705, 707, 774, 790, 804, 820, 837, 855, 879, 890

TABLE S3

Trait	Sex	
	Female	Male
Development time	$F_{1,1675} = 7.962$** $R^2 = 0.004$	$F_{1,1699} = 1.112$ $R^2 = 0.0007$
Wing area	$F_{1,739} = 168.5$*** $R^2 = 0.1846$	$F_{1,742} = 143.5$*** $R^2 = 0.1609$
Wing loading	$F_{1,738} = 137.3$*** $R^2 = 0.1557$	$F_{1,742} = 133.5$*** $R^2 = 0.1514$
Starvation tolerance	$F_{1,1115} = 83.7$*** $R^2 = 0.0689$	$F_{1,1098} = 57.54$*** $R^2 = 0.0489$

* $p < 0.05$; ** $p < 0.01$; *** $p < 0.001$.

```
import sys
import collections
import math
from math import sqrt
from optparse import OptionParser,OptionGroup

#version 1.0
#Author: Martin Kapun

#########################################################   HELP
######################################################################
###
#print
usage=""
parser = OptionParser(usage=usage)
group=OptionGroup(parser,"""
H E L P s:
_________

Description:

        """)

#########################################################   CODE
######################################################################
###

parser.add_option("-d", "--dgrp", dest="d", help="dgrp data")
parser.add_option("-r", "--ref", dest="r", help="lines containing the
reference allele separated by a ','; SetA and Set B lines need to be
separated by a '+'")
parser.add_option("-a", "--alt", dest="a", help="lines containing the
alternative allele separated by a ','; SetA and Set B lines need to be
separated by a '+'")
parser.add_option("-o", "--out", dest="o", help="output file")
parser.add_option("-l", "--label", dest="l", help="label for IGV file
in IGViewer",default="FST")
parser.add_option_group(group)
(options, args) = parser.parse_args()

##  some functions

def lineparse(data,ln):
    """ parse a single line of the freeze2 SNP dataset of the DGRP
lines """
    import collections
    a=data.rstrip().split()
    CHROM,POS,ID,REF,ALT,QUAL,FILTER,INFO,FORMAT=a[:9]
    allelecode={"0":REF}
    # exclude indels
    if len(ALT.split(","))>1:
        return "NA"
    if len(ALT)>1 or len(REF)>1:
        return "NA"

    for i in range(1,len(ALT.split(","))+1):
        allelecode[str(i)]=ALT.split(",")[i-1]
    lines=a[9:]
    allelecounts=collections.defaultdict(list)
    for i in range(len(lines)):
        if i not in ln:
            continue
        for j in lines[i].split("/"):
            allelecounts[i].append(allelecode[j])
    return CHROM,POS,allelecounts

def pi(x):
    """ calculate pi on a SNP-wise basis. where x is a vector of all
allelefreqs and n is the samplesize)"""
    if x==[]:
        return "nan"
    else:
        corr=(len(x)-1)/float(len(x))
        freqsum=sum([y**2 for y in x])
        return (1-freqsum)*corr

def af(allelecounts,ln,lin):
    linestr=""
    afsf=collections.defaultdict(int)
    afs2=collections.defaultdict(float)
    #print allelecounts
    counts=0
    for k,v in allelecounts.items():
        if k not in ln:
            continue
        linestr+=str(lin[k])
        linestr+=":"
        for alle in v:
            afsf[alle]+=1
            counts+=1.0
            linestr+=alle
            linestr+=","
    #print afs,counts
    #print "af:",[x/counts for x in afs.values()]
    for k,v in afsf.items():
        afs2[k]=v/counts

    return afs2,linestr
```

```
################################### CODE
#########################################

# read data
dgrpdata=open(options.d,"r")
ref=options.r.split("+")
alt=options.a.split("+")
out=open(options.o,"w")
label=options.l

# define containers
linepos_hash=collections.defaultdict(lambda:collections.defaultdict(li
st))
lineID_hash=collections.defaultdict(lambda:collections.defaultdict(lis
t))
out.write("Chromosome\tStart\tEnd\tfeature\tFST\n")
count=1

## make position hash for all datasets
for i in range(len(ref)):
    lineID_hash["ref"][i]=map(int,ref[i].split(","))
for i in range(len(alt)):
    lineID_hash["alt"][i]=map(int,alt[i].split(","))

linehash=collections.defaultdict(int)
positions=[]
## loop through all lines in SNP dataset
for l in dgrpdata:
    if count%100000==0:
        print count,"SNPs processed"
    count+=1
    if l.startswith("##"):
        continue
    if l.startswith("#"):
        linenames=map(int,[x.replace("DGRP-","") for x in l.split()
[9:]])
        for i in range(len(linenames)):
            linehash[linenames[i]]=i
        for allele,replicates in lineID_hash.items():
            for replicate,lines in replicates.items():
                for line in lines:
                    if line not in linehash:
                        continue
                    linepos_hash[allele]
[replicate].append(linehash[line])
                    if linehash[line] not in positions:
                        positions.append(linehash[line])
        continue
    linedata=lineparse(l,positions)
    if linedata=="NA":
        continue
    chr, pos, data=linedata
    fstlist=[]
    for i in range(len(linepos_hash["ref"])):
        for j in range(len(linepos_hash["ref"])):
            lines1=linepos_hash["ref"][i]
            lines2=linepos_hash["alt"][j]
            linestot=lines1+lines2
            af1=af(data,lines1,linenames)
            af2=af(data,lines2,linenames)
            aftot=af(data,linestot,linenames)
            p1=pi(af1[0].values())
            p2=pi(af2[0].values())
            ptot=pi(aftot[0].values())
            if ptot==0:
                fst="nan"
                continue
            elif "nan" in [p1,p2,ptot]:
                fst="nan"
            else:
                fst=(ptot-((p1+p2)/2))/ptot
            if fst<0:
                fst=0
            fstlist.append(fst)
    if fstlist==[]:
        continue
    out.write("\t".join(map(str,
[chr,int(pos)-1,pos,label,sum(fstlist)/len(fstlist)]))+"\n")
```

```
#!/bin/sh

#  FST.sh
#
#
#  Created by Martin Kapun on 25.06.18.
#

### calculating Weir & Cockerham (1984) FST among two sets of lines in
the DGRP dataset
### the DGRP lines need to be provided by a comma-separated list of
lines in plain numbers (i.e. without the "Ral" prefix)
### The output is in IGV file format (http://
software.broadinstitute.org/software/igv/IGV)

python fst_dgrp.py \
--dgrp freeze2.bins_sorted.vcf \
--ref
59,138,181,208,228,320,332,338,357,374,426,437,443,491,508,509,535,554
,555,595 \
--alt
21,93,149,217,227,235,237,239,272,287,309,350,352,358,359,362,381,386,
392,406 \
--out > output.igv
```