Optimizing rabies vaccination

Kim Cuddington

04/07/2023

The following .Rmd file contains code to reproduce the analysis in Cuddington & McAuliffe. Optimizing rabies vaccination of dogs in India.

The code also requires a spreadsheet from the supplemental materials of Gibson, A. D., Wallace, R. M., Rahman, A., Bharti, O. K., Isloor, S., Lohr, F., ... & Day, M. J. (2020). Reviewing solutions of scale for canine rabies elimination in India. Tropical medicine and infectious disease, 5(1), 47.("Bang Scen A - OBH - 11d.xlsx").

All figures are sent to .jpeg files

Assign parameter values and read in cost data

```
rm(list = ls())
library(lpSolve)
library(openxlsx)

# Probabilty of vaccination method reaching dogs of each category Gibson et al.
# (2020) values for Bangalore
CP = c(0.95, 0.8, 0.05)
DD = c(0.95, 0.8, 0.1)
CVR = c(0.05, 0.7, 0.64)
ORV = c(0.05, 0.95, 0.79)

# vaccine efficacy Gibson et al. (2020) for Bangalore
Inj = 1
Oral = 0.8

# range of proportion of NC and cost of oral bait vaccine
NCseq = seq(from = 0.05, to = 0.99, by = 0.01)
obs = seq(from = 0.5, to = 4.5, by = 0.05)
```

```
# Gibson, A. D., Wallace, R. M., Rahman, A., Bharti, O. K., Isloor, S., Lohr, F.,
# ... & Day, M. J. (2020). Reviewing solutions of scale for canine rabies
# elimination in India.  Tropical medicine and infectious disease, 5(1), 47.


# Scenarios for dog population size in Bangalore city were calculated based on
# best, mean and worse case scenarios for available dog-to-human ratios in Indian
# urban settings; Scenario A 83:1, (B) 50:1, (C) 23:1
```

```
# there are separate spreadsheets for each of these supplementary materials
# https://www.mdpi.com/2414-6366/5/1/47

# For Wallace sheets
# rateG=read.xlsx(cosfile,colNames=FALSE,sheet=4,rows=c(5:8),cols=c(1:3)) these
# are NOT the same on the Gibson sheet

costfile = "Bang Scen A - OBH - 11d.xlsx"  #name of excel sheet with costs

# read in costs
costGm = read.xlsx(costfile, sheet = 4, rows = c(28:73), cols = c(1:7), startRow = 28)

# remove missing lines
costGm = costGm[complete.cases(costGm), ]

colnames(costGm)[4:6] = c("low_unit_cost", "mean_unit_cost", "hi_unit_cost")
rownames(costGm) = c(1:nrow(costGm))
costG = as.data.frame(costGm)
costG[, 2:6] = apply(costG[, 2:6], 2, function(x) as.numeric(x, na.rm = TRUE))
```

```
## Warning in FUN(newX[, i], ...): NAs introduced by coercion

## Warning in FUN(newX[, i], ...): NAs introduced by coercion
```

```
# print out
costGtable = costG[-c(1, 2, 9, 10, 13, 14:18), -(2:3)]
rownames(costGtable) = c(1:nrow(costGtable))
knitr::kable(costGtable, digits = 2, caption = "Vaccination costs Gibson et al. 2020")
```

Table 1: Vaccination costs Gibson et al. 2020

Item	low_unit_cost	mean_unit_cost	hi_unit_cost
Vaccination supervisor (1 per 25,000 dogs)	8.00	15.00	20.00
Central Point technician	6.00	12.00	14.00
Door to Door technician	7.00	13.00	16.00
Capture/Vax/Release technician	7.00	11.00	13.00
ORV technician	7.00	13.00	16.00
Driver	5.00	8.00	10.00
Other vehicle (ie rental, purchase, other)	10.00	15.00	20.00
Gasoline	10.00	15.00	20.00
Coolers	6.00	10.00	15.00
Dog handling (e.g., muzzles)	20.00	40.00	60.00
CVR Kit	40.00	60.00	80.00
First-aid	5.00	7.00	10.00
Central Vaccine Storage	20.00	30.00	40.00
CP/DD Bite PEP (1 in 2,000)	60.00	100.00	140.00
CVR Bite Booster PEP (1 in 500)	60.00	100.00	140.00
Other equipment / supplies	2000.00	2500.00	3000.00
Vaccines (Parenteral)	0.30	0.40	0.50
Syringes and needles	0.11	0.13	0.15
Vaccination certificates	0.03	0.05	0.08
Dog marking	0.02	0.03	0.04

Calculate per dog costs for optimization

```
# daily vaccination rates for each method N.B. Gibson 2019 Overall staff rate
# (dogs/person/day) OBH=41 CVR=14.6 DD=23.9

# use vax rate per team per day from Gibson 2020 sheets
rateG = read.xlsx(costfile, colNames = FALSE, sheet = 6, rows = c(33:36), cols = c(1:2))

rateG[4, 2] = 30  #uncomment to make ORV rate the same as the others

# costs for parenteral vaccines
injvaxcost = sum(costG[c(27:29), 5])

# estimate of the number of vaccination staff required per vax method, then
# divide by per person day vax rate
numpeople = c(1, 2, 4, 2) * (costG[4:7, 5])/rateG[, 2]

# costs between van and cycle not distinguished by Gibson; assume 2x
vehrent = sum(costG[11:12, 5])/rateG[, 2]
vehrent = vehrent * c(0, 1, 2, 1)

# specialized costs (so exclude coolers here) for example CVR kit...
cvrkit = costG[21, 5]/20000  #estimate on the per dog cost based on 20000 vaxs
cvrpep = costG[25, 5]/500  # given in Gibson et al 2020
cvrdriver = (costG[8, 5])/rateG[3, 2]
CVRcost = costG[25, 5]/500 + (costG[8, 5] * 8)/rateG[3, 2]  #cost without kit

CPDDcost = costG[24, 5]/2000  # given in Gibons et al. 2020
OBcost = costG[24, 5]/1000  #assume higher than DD/CP, but lower than CVR

# construct vector of equip costs
equip = c(CPDDcost, CPDDcost, CVRcost, OBcost)

# construct vector of staff costs
techs = paste0(costG[4:7, 1], "s")
techs[1] = paste(techs[1], "(1 per dog)")
techs[2] = paste(techs[2], "(2 per dog)")
techs[3] = paste(techs[3], "(4 per dog)")
techs[4] = paste(techs[4], "(2 per dog)")

# table of per dog costs
percost = c(costG[27:31, 5], numpeople, cvrdriver, vehrent[-1], CPDDcost, cvrpep,
    OBcost)
peritem = c(costG[27:28, 1], paste(costG[29, 1], "(CP/DD/CVR)"), paste(costG[30,
    1], "(CP/DD)"), paste(costG[31, 1], "(CVR/ORV)"), techs, "CVR driver", "DD vehicle rental & fuel",
    "CVR vehicle rental & fuel", "ORV vehicle rental & fuel", costG[24, 1], costG[25,
        1], "ORV PEP (1 in 1,000)")

# table of fixed costs, we're not including these since they should not vary with
# methods
staff = costG[c(1:3, 9), 5]
staffl = paste(costG[c(1:3, 9), 1], "per day")
media = costG[c(15:16), 5]
```

```
medial = c(paste0(costG[15, 1], "10000 per 25000 dogs"), paste0(costG[16, 1], "(1 per 25000 dogs"))

equipF = costG[c(17:19, 20:23), 5]
equipFl = c(costG[17, 1], paste(costG[18, 1], "(1 in 6 CP techs)"), paste(costG[19,
    1], "(1 per 2 techs)"), paste(costG[20, 1], "(1 per 2 CVR techs)"), paste(costG[21,
    1], "(1 per CVR tech)"), paste(costG[22, 1], "(1 per 2 techs)"), costG[23, 1])


fcost = c(staff, media, equipF)
fcostitem = c(staffl, medial, equipFl)
fixtab = data.frame(item = fcostitem, cost = fcost)
knitr::kable(fixtab, digits = 2, caption = "Fixed cost for a vaccination campaign using data from Gibso
    col.names = c("Item", "Fixed costs Cost (USD)"))
```

Table 2: Fixed cost for a vaccination campaign using data from Gibson et al. (2020)

Item	Fixed costs Cost (USD)
Program manager per day	18.0
Informational supervisor per day	18.0
Vaccination supervisor (1 per 25,000 dogs) per day	15.0
Other Personnel per day	8.0
Media (e.g. posters)10000 per 25000 dogs	0.6
Air time (radio, car with speakers, etc.)(1 per 25000 dogs	35.0
Other costs	5000.0
Tables (1 in 6 CP techs)	20.0
Coolers (1 per 2 techs)	10.0
Dog handling (e.g., muzzles) (1 per 2 CVR techs)	40.0
CVR Kit (1 per CVR tech)	60.0
First-aid (1 per 2 techs)	7.0
Central Vaccine Storage	30.0

```
pertab = data.frame(item = peritem, cost = percost)
knitr::kable(pertab, digits = 2, caption = "Per dog vaccination costs calculated using data from Gibson
    col.names = c("Item", "Cost per dog (USD)"))
```

Table 3: Per dog vaccination costs calculated using data from Gibson et al. (2020)

Item	Cost per dog (USD)
Vaccines (Parenteral)	0.40
Syringes and needles	0.13
Vaccination certificates (CP/DD/CVR)	0.05
Dog marking (CP/DD)	0.03
NA (CVR/ORV)	NA
Central Point technicians (1 per dog)	0.40
Door to Door technicians (2 per dog)	0.87
Capture/Vax/Release technicians (4 per dog)	1.47
ORV technicians (2 per dog)	0.87
CVR driver	0.27

Item	Cost per dog (USD)
DD vehicle rental & fuel	1.00
CVR vehicle rental & fuel	2.00
ORV vehicle rental & fuel	1.00
CP/DD Bite PEP (1 in 2,000)	0.05
CVR Bite Booster PEP (1 in 500)	0.20
ORV PEP (1 in 1,000)	0.10

```
# do inlcude cost of ORV info campaign since will differ between campaigns with
# and without obmedia=sum(c(costG[c(15:16), 5]*c(10000, 30))) alternate
# calculation
obmedia = 10000

# material costs for injection
cpcost = injvaxcost
ddcost = injvaxcost
cvrcost = injvaxcost + costG[30, 5] - costG[29, 5]
```

Varying accessibility

I've displayed various elements regarding varying accessibility as text (since we cannot loop over chunks in the .Rmd compile even if eval=FALSE, and I have been using the chunk option eval to control looping).

This material should be added back as code. However, it is rather slow and many users may not wish to do so.

use this setup to change vaccine accessibility

' comply=seq(from=0, to=1.0, by=0.005)

st_comply=data.frame(cp=NA, dd=NA, cvr=NA, orv=NA, comp=NA)

pc_OVR=array(NA, dim=c(length(NCseq), length(obs), length(comply))) pc_DD=array(NA, dim=c(length(NCseq), length(obs), length(comply))) pc_CP=array(NA, dim=c(length(NCseq), length(obs), length(comply))) pc_CVR=array(NA, dim=c(length(NCseq), length(obs), length(comply)))

for (zq in seq_along(comply)){ print(zq)

CP=c(.95,comply[zq],.05) #lower CP compliance for SC dogs in Bangalore `#DD=c(.95,.8,.1)`#CVR=c(.05,.70,.64) '#ORV=c(.05,.95,comply[zq]) #lower OVR accessibility for some reason (bait palatability)

Set static accessibility values below

```
pve = data.frame(CP, DD, CVR, ORV)
vaxcats = colnames(pve)
dogcats = c("C", "SC", "NC")
rownames(pve) = dogcats

knitr::kable(pve, digits = 2, caption = "Vaccination accessibility from Gibson et al. 2020")
```

Table 4: Vaccination accessibility from Gibson et al. 2020

	CP	DD	CVR	ORV
C	0.95	0.95	0.05	0.05
SC	0.80	0.80	0.70	0.95
NC	0.05	0.10	0.64	0.79

```
# Net vaccine effectiveness
pve = t(t(pve) * c(Inj, Inj, Inj, Oral))

# total size of dog population, either vector to produce fig 3, or single number
# Dsize=seq(from=5000, to=150000, by=5000)
Dsize = 50000  #total size of dog population
Dt = Dsize

# use this switch for either 1= vaccine coverage constraint on sum of SC and NC
# dogs, or 2= seperate coverage constraints on SC and NC dogs, or 3= seperate,
# but probabilistic constraints
iq = 2

# percentile for probabilistic constraints (note that 0.5 is the same as the
# mean) we investigated 0.2 and 0.7
probs = 0.7

bkevs = vector()
bkevm = matrix(NA, nrow = length(NCseq), ncol = length(Dsize))
slvst = matrix(NA, nrow = length(obs), ncol = length(NCseq))
tstaff = matrix(NA, nrow = length(obs), ncol = length(NCseq))
tcost = matrix(NA, nrow = length(obs), ncol = length(NCseq))
tdogs = matrix(NA, nrow = length(obs), ncol = length(NCseq))
```

```
for (jp in seq_along(Dsize)) {
    Dt = Dsize[jp]

    # for (iq in 1:2){ #uncomment to run two scenarios one for 20% success and one
    # for 70% find solutions for varying NC and oral bait cost

    svsoln = list()
    pvex = list()
    pdogcost = matrix(NA, nrow = length(obs), ncol = length(NCseq))
    cnt = 0


    for (qk in seq_along(NCseq)) {
        # for every proprotion of NC dogs
        NC = NCseq[qk]
        C = (1 - NC)/2
        SC = C
        if (SC < 0)
            SC = 0
```

```
pC = c(C, SC, NC)
d = Dt * pC
prb = probs

for (q in seq_along(obs)) {
    # for every cost of an individual oral bait
    cnt = cnt + 1
    obcost = obs[q] + costG[30, 5]

    # set up cost matrix
    costvax = c(cpcost, ddcost, cvrcost, obcost)
    bcost = costvax + numpeople + equip + vehrent

    # number of vax attempts per each successful vax
    pvecd = 1/pve
    pvecd[, ] = 1
    # cost per successful vax attempt cost per vax attempt
    pvecd = bcost * t(pvecd)
    pvecdd = pvecd

    # set up objective matrix
    obj = c(as.matrix((pvecdd)))
    m <- 3
    n <- 4
    constr <- matrix(0, n + m, n * m)
    for (i in 1:m) {
        for (j in 1:n) {
          constr[i, n * (i - 1) + j] <- 1
          constr[m + j, n * (i - 1) + j] <- 1
        }
    }

    # add constraints for combined coverage constraint
    if (iq == 1) {
        cpcon = constr[1:3, ]
        constr = rbind(cpcon, constr[2, ] + constr[3, ])
        constr[1, 1:4] = as.matrix(pve[1, ])
        constr[2, 5:8] = as.matrix(pve[2, ])
        constr[3, 9:12] = as.matrix(pve[3, ])
        constr[2, ] = constr[2, ] + constr[3, ]
        constr = constr[1:2, ]
        constr = rbind(constr, cpcon)
        rhs <- c(0.7 * d[1], 0.7 * (d[2] + d[3]), d[1] * 1, d[2] * 1, d[3] *
          1)
        constr.dir <- c(rep(">=", 2), rep("<=", 3))
    }

    # set seperate coverage constraints
    if (iq > 1) {
        cpcon = constr[1:3, ]
        constr = rbind(cpcon, cpcon)
        constr[1, 1:4] = as.matrix(pve[1, ])
        constr[2, 5:8] = as.matrix(pve[2, ])
```

```
    constr[3, 9:12] = as.matrix(pve[3, ])
    rhs <- c(0.7 * d[1], 0.7 * d[2], 0.6 * d[3], d[1] * 1, d[2] * 1,
      d[3] * 1)
    constr.dir <- c(rep(">=", 3), rep("<=", 3))

    # set probabilitist coverage constraints
    if (iq == 3) {
      qp = vector()
      qp[1] = qnorm(prb)
      qp[2] = qnorm(prb)
      qp[3] = qnorm(prb)
      vp = 0.1
      rhs <- c(0.7 * d[1] + vp * 0.7 * d[1] * qp[1], 0.7 * d[2] + vp *
        0.7 * d[2] * qp[2], 0.6 * d[3] + vp * 0.6 * d[3] * qp[3], d[1] *
        1, d[2] * 1, d[3] * 1)
    }

}


# find optimal solution
prod.trans <- lp("min", objective.in = obj, constr, constr.dir, rhs)
rm(soln)
slvst[q, qk] = prod.trans$status
soln = (matrix(prod.trans$solution, nrow = 4))
# print(soln) print(obcost)


pvex = soln * t(pve)
pvex = pvex * c(Inj, Inj, Inj, Oral)
solntab = (rbind(soln, colSums(pvex)/d))
solntab = rbind(solntab, d)
colnames(solntab) = dogcats
rownames(solntab) = c(vaxcats, "%vax", "dogs")
svsoln[[cnt]] = soln


knitr::kable(solntab, digits = 2, caption = "Table: Optimal vaccine allocation strategy")
campcost = soln * bcost


# Calc needed person/days

meths = solntab[1:4, ]
methunits = rowSums(meths)
pdays = methunits/rateG[, 2]

# Calc needed persons (/by campaign days) one month?

staff = pdays/30
# tstaff=sum(staff)
tstaff[q, qk] = sum(staff)
pdogcost[q, qk] = (sum(campcost) + obmedia)/sum(meths)
```

```
        tcost[q, qk] = sum(campcost) + obmedia
        tdogs[q, qk] = sum(meths)
    }

}
if (iq == 1)
    pdogcost1 = pdogcost
if (iq > 2)
    pdogcost2 = pdogcost



# to check the cost impact of oral baits set efficicacy to zero and rerun the
# analysis without three vaccination methods

svsolnNO = list()
pvexNO = list()
slvstNO = rep(NA, length(NCseq))
pdogcostNO = rep(NA, length(NCseq))
tstaffNO = rep(NA, length(NCseq))
cnt = 0
for (qk in seq_along(NCseq)) {
    cnt = cnt + 1

    NC = NCseq[qk]
    C = (1 - NC)/2
    SC = C

    if (SC < 0)
        SC = 0
    pC = c(C, SC, NC)
    d = Dt * pC

    pveno = pve
    pveno = pveno[, -4]  #omit ORV
    pveno

    pvecdno = 1/pveno
    pvecdno[, ] = 1
    # cost per successful vax attempt
    pvecdno = bcost[-4] * t(pvecdno)
    objno = c((as.matrix(pvecdno)))

    m <- 3
    n <- 3
    constr <- matrix(0, n + m, n * m)
    for (i in 1:m) {
        for (j in 1:n) {
            constr[i, n * (i - 1) + j] <- 1
            constr[m + j, n * (i - 1) + j] <- 1
        }
    }
    # this array will ensure we cannot exceed the number of dogs in a category
```

```
# set constraints for combined solution
if (iq == 1) {
    cpcon = constr[1:3, ]
    constr = rbind(cpcon, constr[2, ] + constr[3, ])


    constr[1, 1:3] = as.matrix(pveno[1, ])
    constr[2, 4:6] = as.matrix(pveno[2, ])
    constr[3, 7:9] = as.matrix(pveno[3, ])
    constr[2, ] = constr[2, ] + constr[3, ]
    constr = constr[1:2, ]
    constr = rbind(constr, cpcon)

    rhs <- c(0.7 * d[1], 0.7 * (d[2] + d[3]), d[1] * 1, d[2] * 1, d[3] *
        1)
    constr.dir <- c(rep(">=", 2), rep("<=", 3))
}

# set separate constraints
if (iq > 1) {

    cpcon = constr[1:3, ]
    constr = rbind(cpcon, cpcon)
    constr[1, 1:3] = as.matrix(pveno[1, ])
    constr[2, 4:6] = as.matrix(pveno[2, ])
    constr[3, 7:9] = as.matrix(pveno[3, ])
    rhs <- c(0.7 * d[1], 0.7 * d[2], 0.6 * d[3], d[1] * 1, d[2] * 1, d[3] *
        1)

    constr.dir <- c(rep(">=", 3), rep("<=", 3))

    # set probabilistic constraints
    if (iq == 3) {
        qp = vector()
        qp[1] = qnorm(prb)
        qp[2] = qnorm(prb)
        qp[3] = qnorm(prb)
        rhs <- c(0.7 * d[1] + vp * 0.7 * d[1] * qp[1], 0.7 * d[2] + vp *
          0.7 * d[2] * qp[2], 0.6 * d[3] + vp * 0.6 * d[3] * qp[3], d[1] *
          1, d[2] * 1, d[3] * 1)
    }

}

# find optimial solution
prod.trans <- lp("min", objective.in = objno, constr, constr.dir, rhs)

rm(soln)  #remove old solution when looping
soln = (matrix(prod.trans$solution, nrow = 3))
svsolnNO[[cnt]] = soln
slvstNO[[cnt]] = prod.trans$status

pvex = soln * t(pveno)
```

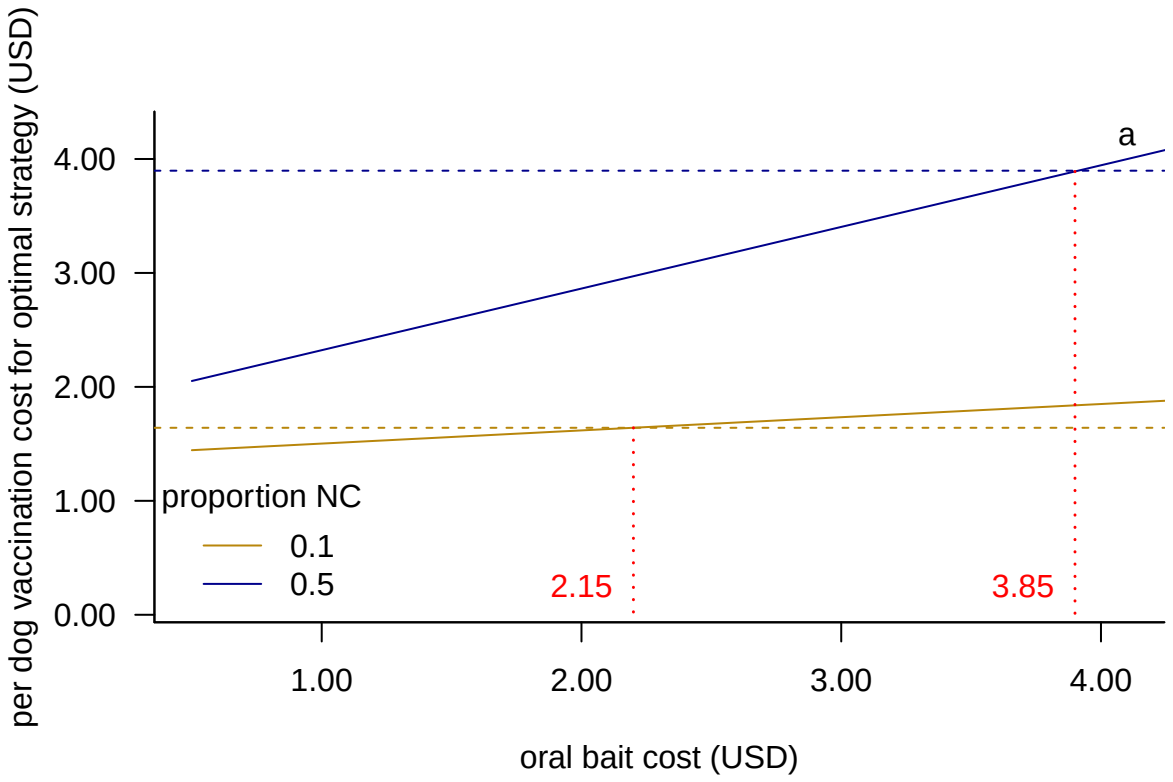
```
        solntab = (rbind(soln, colSums(pvex)/d))
        solntab = rbind(solntab, d)
        colnames(solntab) = dogcats
        rownames(solntab) = c(vaxcats[-4], "%vax", "dogs")
        solntab


        knitr::kable(pvecdd, digits = 2, caption = "Table: no oral baits: Vax costs * 1/efficacy: cost p
        knitr::kable(solntab, digits = 2, caption = "Table: No oral baits: Optimal vaccine allocation s

        campcostNO = soln * bcost[1:3]


        # Calc needed person/days

        meths = solntab[1:3, ]
        methunits = rowSums(meths)
        pdays = methunits/rateG[1:3, 2]

        # Calc needed persons (/by campaign days) one month?

        staff = pdays/30
        tstaffNO[qk] = sum(staff)



        pdogcostNO[qk] = sum(campcostNO)/sum(meths)
    }
}
```

Find optimal solutions with and without oral baits

```
## Warning in rm(soln): object 'soln' not found
```

```
# plot an example of the difference in per dog costs produce fig 1
# jpeg('baseR_figure1.jpeg', width =180, height = 150, units = 'mm', pointsize =
# 12, quality = 75, res = 300)


sprlist = c(6, 46)
spr = sprlist[1]
plot(NA, xaxt = "n", yaxt = "n", xlab = "oral bait cost (USD)", ylab = "per dog vaccination cost for op
    type = "l", lwd = 2, las = 1, bty = "L", xlim = c(0.5, 4.1), ylim = c(0.1, 4.25))
# main=paste('Scenario with %NC dogs =', NCseq[spr])) # no default y axis
p <- pretty(par("usr")[3:4])  # find nice breaks at actual y range
q <- pretty(par("usr")[1:2])
l <- formatC(p, format = "f", digits = 2)  # format w/12
m <- formatC(q, format = "f", digits = 2)
axis(2, at = p, labels = l, las = 1)
axis(1, at = q, labels = m)
clrs = c("dark goldenrod", "dark blue")
for (kl in 1:2) {
```

```
    spr = sprlist[kl]

    lines(obs, pdogcost[, spr], col = clrs[kl])
    cls = c("goldenrod", "dark green")
    abline(h = pdogcostNO[spr], col = clrs[kl], lty = 2)
    idx = which.min(round(abs(pdogcost[, spr] - pdogcostNO[spr]), 2))


    segments(obs[idx], y0 = pdogcost[idx, spr], x1 = obs[idx], y1 = 0, col = "red",
        lty = 3, lwd = 1.5)
    text(x = obs[idx] - 0.2, y = 0.25, paste0(sprintf("%.2f", round(obs[idx - 1],
        2))), col = "red")
}


legend("bottomleft", legend = NCseq[sprlist], bty = "n", col = c("dark goldenrod",
    "dark blue"), lty = 1, title = "proportion NC")


# legend('bottomleft', legend=c('with ORV', 'no ORV'), bty='n', col=1,
# lty=c(1,2), title='campaign type')

text(x = 4.1, y = 4.2, "a")
```



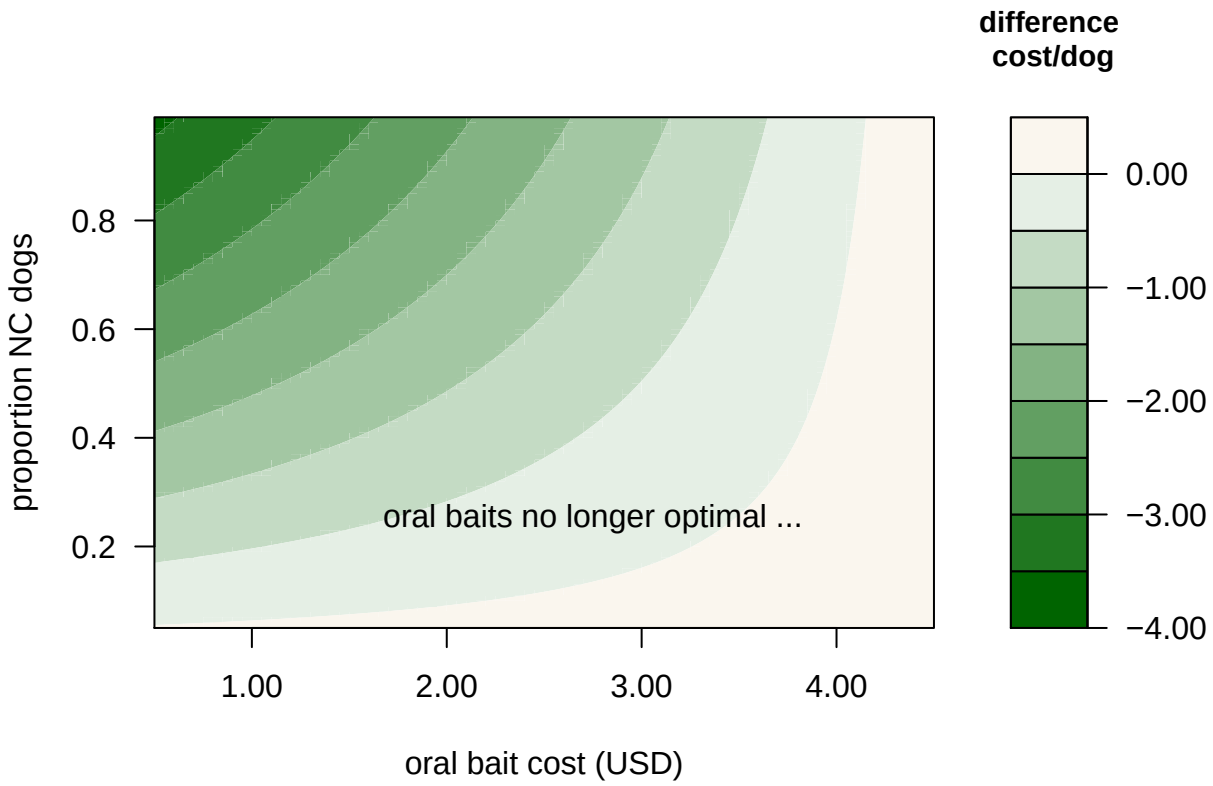
```
# dev.off()
```

```
if (idx==1){
  bkevs[jp]=NA
  }else{
bkevs[jp]=obs[idx-1]
}
```

```
M=sweep(pdogcost,2, pdogcostNO)
M=round(M, 4)

indxs=apply(M, 2, function(x) which.min(abs(x)))
bkevm[,jp]=obs[indxs]

qm=seq(-4, 3)
qml=c("-4.00","-3.00","-2.00","-1.00","0.00","1.00", "2.00", "3.00" )
clr=colorRampPalette(c("dark green", "white", "dark goldenrod"),bias=.50)(12)

Mx=M
if (iq==1){
  # uncomment for case where there is no solution
#maxM=which(slvstNO>0, arr.ind=TRUE)[1]
#maxMO=which(slvst>0, arr.ind=TRUE)[1,2]
#Mx[,maxM:ncol(M)]=NA
}

#produce figure 2
#jpeg('baseR2_figure3.jpeg', width =180,
 #    height = 150,
  #   units = 'mm', pointsize = 12,
   #   quality = 75,
    # res = 300)
filled.contour(obs,NCseq, Mx,
col=clr,nlevels=12,
plot.title = title(main = "",
    ylab = "proportion NC dogs", xlab = "oral bait cost (USD)",
    # abline(v=bkline),
    text(x=2.75, y=0.25,paste("oral baits no longer optimal",sprintf('\u2192')))),
    key.title = {par(cex.main=0.9);title(main = "difference\n cost/dog")},
    key.axes=axis(side=4,at=qm,labels=qml),
    plot.axes={
      #abline(v=3.6)
      axis(2, at=c(0.2, 0.4, 0.6, 0.8), labels=c("0.2", "0.4", "0.6", "0.8"),las=1)
axis(side=1,at=q[2:5],labels=m[2:5])
      }
)
```

```
## Warning in text.default(x = 2.75, y = 0.25, paste("oral baits no longer
## optimal", : conversion failure on 'oral baits no longer optimal →' in
## 'mbcsToSbcs': dot substituted for <e2>
```

```
## Warning in text.default(x = 2.75, y = 0.25, paste("oral baits no longer
## optimal", : conversion failure on 'oral baits no longer optimal →' in
## 'mbcsToSbcs': dot substituted for <86>

## Warning in text.default(x = 2.75, y = 0.25, paste("oral baits no longer
## optimal", : conversion failure on 'oral baits no longer optimal →' in
## 'mbcsToSbcs': dot substituted for <92>

## Warning in text.default(x = 2.75, y = 0.25, paste("oral baits no longer
## optimal", : font metrics unknown for Unicode character U+2192
```




```
#dev.off()
```

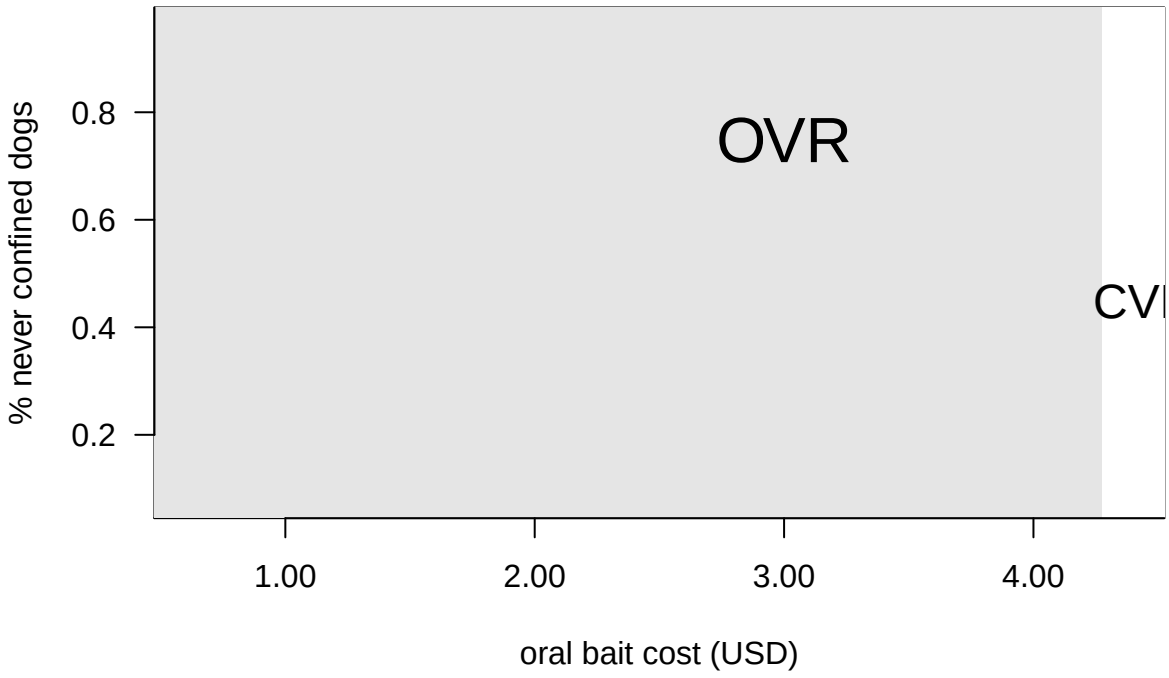
```
# print('extract!')
cname = as.vector(outer(rownames(pvecd), colnames(pvecd), paste, sep = "_"))

solnf <- as.data.frame(matrix(ncol = 12, nrow = length(svsoln)))
names(solnf) <- cname
for (i in 1:12) {
    solnf[, i] = sapply(svsoln, `[`, i)
}

plen = vector()
Nlen = vector()
```

```
cnt = 0
for (i in seq_along(NCseq)) {
    for (j in seq_along(obs)) {
        cnt = cnt + 1
        Nlen[cnt] = NCseq[i]
        plen[cnt] = obs[j]

    }
}
```

```
solnf$clustapriori = ifelse(solnf$ORV_NC == 0, 1, ifelse(solnf$CVR_NC > 0, 3, 2))
ORVclustsurf = matrix(as.numeric(solnf$clustapriori), ncol = length(NCseq), nrow = length(obs))
```

```
#jpeg('baseR3_figure.jpeg', width =180,
#     height = 150,
#     units = 'mm', pointsize = 12,
 #    quality = 75,
 #    res = 300)

image(obs, NCseq, ORVclustsurf, col = c("white","grey40", "grey90"),
axes=F, frame=TRUE,bty="o",
    ylab = "% never confined dogs", xlab = "oral bait cost (USD)"

)

      #abline(v=3.6)
      axis(2, at=c(0.2, 0.4, 0.6, 0.8, 1.0), labels=c("0.2", "0.4", "0.6", "0.8", "1.0"),las=1)
axis(side=1,at=q[2:6],labels=m[2:6])
 text(3., 0.75, "OVR", cex=2.)
  text(4.55, 0.45, paste("CVR",sprintf('\u2192')), cex=1.5)
```

```
## Warning in text.default(4.55, 0.45, paste("CVR", sprintf("→")), cex = 1.5):
## conversion failure on 'CVR →' in 'mbcsToSbcs': dot substituted for <e2>

## Warning in text.default(4.55, 0.45, paste("CVR", sprintf("→")), cex = 1.5):
## conversion failure on 'CVR →' in 'mbcsToSbcs': dot substituted for <86>

## Warning in text.default(4.55, 0.45, paste("CVR", sprintf("→")), cex = 1.5):
## conversion failure on 'CVR →' in 'mbcsToSbcs': dot substituted for <92>

## Warning in text.default(4.55, 0.45, paste("CVR", sprintf("→")), cex = 1.5): font
## metrics unknown for Unicode character U+2192
```




```
#dev.off()
```

```
# set eval = TRUE to produce plots for varying total pop size

print(Dsize)
print(m)

# plots for total dog population size
jpeg("baseR3_figure.jpeg", width = 180, height = 150, units = "mm", pointsize = 12,
    quality = 75, res = 300)
plot(Dsize, bkevm[1, ], type = "b", pch = 16, cex = 0.8, las = 1, bty = "l", ylim = c(0,
    4.5), xlim = c(0, 150000), yaxt = "n", lwd = 2, ylab = "break-even cost of oral baits (USD)",
    xlab = "total dog population")
# axis(2, at=c(0.2, 0.4, 0.6, 0.8), labels=c('0.2', '0.4', '0.6', '0.8'),las=1)
axis(side = 2, at = q[1:7], labels = m[1:7], las = 2)
points(Dsize, bkevm[6, ], type = "b", pch = 16, cex = 0.8, col = "dark goldenrod",
    lwd = 2)
lines(Dsize, bkevm[21, ], type = "b", pch = 16, cex = 0.8, col = 4, lwd = 2)
lines(Dsize, bkevm[91, ], type = "b", cex = 0.8, pch = 16, col = "pink3", lwd = 2)
abline(v = 50000, col = "grey70")
legend("bottomright", legend = NCseq[c(1, 6, 21, 91)], bty = "n", pch = 16, lty = 1,
    col = c("black", "dark goldenrod", "blue", "pink3"), title = "proportion\n NC dogs")
dev.off()


jpeg("baseR5_figure.jpeg", width = 180, height = 150, units = "mm", pointsize = 12,
```

```
    quality = 75, res = 300)
par(mar = c(5, 5, 2, 2))
plot(NCseq, tdogs[36, ], ylim = c(35000, 50000), type = "l", lwd = 2, las = 1, bty = "l",
    ylab = "", cex = 0.9, cex.lab = 1.2, xlab = "percent of NC dogs in the population")
lines(NCseq, tdogs2[36, ], lwd = 2, lty = 2)
lines(NCseq, tdogs7[36, ], lwd = 2, lty = 3)
title(ylab = "total number of vaccinations required", line = 4, cex.lab = 1.2)
legend("bottomright", legend = c(0.7, 0.5, 0.2), title = "prob of success", lty = c(3,
    1, 2), bty = "n")
dev.off()
```

```
# example result

solout = svsoln[[as.integer(length(svsoln)/2)]]
row.names(solout) <- c("CP", "DD", "CVR", "ORV")

knitr::kable(solout, digits = 0, caption = "Example of optimal solution", col.names = c("CP",
    "SC", "NC"))
```

Table 5: Example of optimal solution

	CP	SC	NC
CP	8842	10500	0
DD	0	0	0
CVR	0	0	0
ORV	0	0	24684

Varying accesilbity

Add this material bad to finish looping thru accessiblity

lr=rep(obs,length(NCseq)) lq=rep(NCseq, length(obs)) solnf$ocost = plensolnf$NC=Nlen

cntstrat<-function(x) {sum(x>0)} cntstrat<-function(x) {mean(x)} pc_OVR[, ,zq]=tapply(as.numeric(solnf$ORV_SC), INDE
list(solnf$NC, solnf$ocost), cntstrat)$pc_D D[, , zq]$ = tapply(as.numeric(solnf$DD_SC), INDEX = list(solnf$NC, solnf$ocost),
cntstrat) pc_CVR[, ,zq]=tapply(as.numeric(solnf$CVR_SC), INDEX = list(solnf$NC, solnf$ocost), cntstrat)$pc_C P[, , zq]$ =
tapply(as.numeric(solnf$CP_SC), INDEX = list(solnf$NC, solnf$ocost), cntstrat)

st_comply[zq,]=c(sum(solnf$CP_SC > 0), sum(solnf$DD_SC>0),sum(solnf$CVR_SC > 0), sum(solnf$ORV_SC>0),
comply[zq])

}

}

Varying compliance: figs

```
# figs for varying compliance



matplot(x = st_comply$comp, st_comply[, 1:4]/(length(obs) * length(NCseq)), type = "l",
```

```
    las = 1, lwd = 2, ylab = "percent of solutions including \n a given strategy for SC dogs",
    xlab = "CP compliance rate for SC dogs", ylim = c(-0.1, 1.1), bty = "L", col = c(1,
        2, 3, 5))
legend("right", legend = c("CP", "DD", "CVR", "ORV"), lty = 1:4, col = c(1, 2, 3,
    5), lwd = 2, bty = "n")



iq = 81
filled.contour((pc_CP[iq, , ]), x = obs, y = comply)
filled.contour((pc_OVR[iq, , ]), x = obs, y = comply)


# generate fig 4
par(mfrow = c(1, 1), mar = c(5, 5, 2, 2))
iq = 18
image((pc_CP[iq, , ]), x = obs, y = comply)
image((pc_DD[iq, , ]), x = obs, y = comply)
image((pc_OVR[iq, , ]), x = obs, y = comply)
image((pc_CVR[iq, , ]), x = obs, y = comply)

ipc_CP = ifelse(pc_CP[iq, , ] > 0, 1, 0)
ipc_DD = ifelse(pc_DD[iq, , ] > 0, 50, 0)
ipc_OVR = ifelse(pc_OVR[iq, , ] > 0, 0, 1)
ipc_CVR = ifelse(pc_CVR[iq, , ] > 0, 1, 0)

par(mfrow = c(1, 1))

jpeg("baseR9_figure.jpeg", width = 180, height = 150, units = "mm", pointsize = 12,
    quality = 75, res = 300)

image(ipc_CP, x = obs, y = comply, col = hcl.colors(2, "Oslo", alpha = 0.5, rev = TRUE),
    xlab = "oral bait cost (USD)", ylab = "CP compliance rate for SC dogs", las = 1,
    xlim = c(0.5, 4), axes = F, frame = TRUE, bty = "o")
axis(2, at = c(0.2, 0.4, 0.6, 0.8, 1), labels = c("0.2", "0.4", "0.6", "0.8", "1.0"),
    las = 1)
axis(side = 1, at = 1:4, labels = m[2:5])
image(ipc_DD, x = obs, y = comply, col = hcl.colors(2, "Terrain", alpha = 0.25, rev = TRUE),
    add = TRUE)


image(ipc_OVR, x = obs, y = comply, col = hcl.colors(2, "Blues 3", alpha = 0.25,
    rev = FALSE), add = TRUE)


text(3.25, 0.9, "CP", cex = 1.5)
text(3.25, 0.5, "DD+CP", cex = 1.5)
text(3.25, 0.15, "DD", cex = 1.5)
text(0.75, 0.15, "ORV")
text(0.75, 0.5, "ORV+\nCP")

dev.off()
```