Package 'holdem'

August 28, 2017

Type Package

Title Texas Holdem Simulator

Version 1.2

Date 2017-08-01

Author Frederic Paik Schoenberg,Yixuan Shao

Maintainer Frederic Paik Schoenberg <frederic@stat.ucla.edu>

Description Simulates hands and tournaments of Texas Holdem,
the most popular form of poker.
For examples of probability problems involving Texas Holdem
and a brief explanation of the game see Schoenberg, F. (2011).
An Introduction to Probability with Texas Hold'em Examples.
Taylor and Francis, New York, ISBN-13: 978-1439827680.

Depends graphics, stats

License GPL-2

Repository CRAN

Date/Publication 2017-08-28 10:45:20 UTC

NeedsCompilation no

R topics documented:

`holdem-package` *Simulates Texas Hold'em hands and tournaments*

Description

This package can be used to simulate hands of Texas Holdem and can run Texas Holdem tournaments.

Details

Package:	holdem
Type:	Package
Version:	1.0
Date:	2011-06-15
License:	What license is it under?
LazyLoad:	yes

The most important functions are tourn1(), many1(), deal1(), and handeval(). tourn1() playes a Texas Hold'em tournament between different functions which can bet arbitrary amounts depending on their cards, chips, blinds, etc. many1() plays multiple tournaments and stores the payouts for each player. deal1() deals an individual hand of Texas Hold'em. handeval() figures out who won the hand, when more than one player remains and all the cards are shown.

Author(s)

Frederic Paik Schoenberg

Maintainer: Frederic Paik Schoenberg <frederic@stat.ucla.edu>

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
## out of 1000 hands with 2 players, see how many times
## both players get at least a straight, if neither ever folds.
n = 1000
result = rep(0,n)
for(i in 1:n){
x1 = deal1(2)
b1 = handeval(c(x1$plnum1[1,],x1$brdnum1), c(x1$plsuit1[1,],x1$brdsuit1))
b2 = handeval(c(x1$plnum1[2,],x1$brdnum1), c(x1$plsuit1[2,],x1$brdsuit1))
if(min(b1,b2) > 4000000) result[i] = 1
}
sum(result>.5)

## run a Texas Hold'em tournament between
## 4 players with different strategies, where
## each always goes all in or folds.
name1 = c("gravity","tommy","ursula","vera")
decision1 = list(gravity, tommy, ursula, vera)
tourn1(name1, decision1, myfast1 = 2) ## run quickly
tourn1(name1, decision1, myfast1 = 0) ## run slowly, showing key hands
```

---

bid1 *Preflop bidding*

Description

Runs a round of preflop bidding. Used by tourn1().

Usage

```
bid1(numattable1, playerseats1, chips1, blinds1, dealer1, b3, ntable1, decision1)
```

Arguments

numattable1	Number of players currently remaining at the given table.
playerseats1	list of indices, who's in seat 1, seat 2, etc.
chips1	list of chips left, for players at this table only.

blinds1	vector of (small blind amount, big blind amount).
dealer1	seat that the dealer is in.
b3	cards the players have.
ntable1	how many tables remain in the tournament.
decision1	vector of the functions governing the players' betting

Value

i1	vector indicating who is still in the hand (1) or is out (0)
p1	the size of the pot
p1	the size of the pot
c1	the number of chips everyone has left
rb	the betting for the whole hand
all1	0 if there is more betting in the hand, or 2 if the betting in the hand is all over.
bl1	the betting for the current round
il1	player number indices of who bet
out1	list of who is out, i.e. who folded this round or had folded previously.

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, playerseats1, chips1, blinds1, dealer1, b3, ntable1, decision1){
   round1 = 1
   in1 = 1*(chips1>0.5)
   bet1 = rep(0,numattable1)
   betlist1 = c(0)
   indlist1 = c(0)
   outlist1 = c(0)
   ind1 = dealer1+1
   if(ind1>numattable1) ind1 = 1
   j = 0
   prevbet = 0
   currentbet = 0
   better1 = 0
   board1 = matrix(rep(0,10),ncol=2)
   board1[1:3,1] = b3$brdnum1[1:3]
   board1[1:3,2] = b3$brdsuit1[1:3]
   roundbets = matrix(rep(0,(4*numattable1)),ncol=4)
   pot1 = 0
```

```
    stp = 0
    while(stp < 1){
        myout1 = 0
        j = j+1
        if(j==1){
    bet1[ind1] = min(chips1[ind1],blinds1[1])
    prevbet = bet1[ind1]
    better1 = ind1
    pot1 = bet1[ind1]
    roundbets[ind1,1] = bet1[ind1]
    chips1[ind1] = chips1[ind1] - bet1[ind1]
    betlist1 = c(betlist1,bet1[ind1])
    indlist1 = c(indlist1,ind1)
    outlist1 = c(outlist1,myout1)
        }
        if(j==2){
    bet1[ind1] = min(chips1[ind1],blinds1[2])
    currentbet = max(prevbet,blinds1[2])
    better1 = ind1
    pot1 = pot1 + bet1[ind1]
    roundbets[ind1,1] = bet1[ind1]
    chips1[ind1] = chips1[ind1] - bet1[ind1]
    betlist1 = c(betlist1,bet1[ind1])
    indlist1 = c(indlist1,ind1)
    outlist1 = c(outlist1,myout1)
        }
        if(j==3){better1 = ind1}  # so that big blind can raise!
        if((j > 2.5) && (in1[ind1] > 0.5)){
    crds1 =
  matrix(c(b3$plnum1[ind1,],b3$plsuit1[ind1,]),ncol=2)
    bmax1 = max((bet1[-ind1] + chips1[-ind1])[in1[-ind1]>.5])
    # cat(".a. Seat ",ind1,"'s turn, bmax is ",bmax1,", lastbetter is ",better1,"...\n")
    if(bmax1 < currentbet) currentbet = bmax1
    b1 = round(decision1[[playerseats1[ind1]]](numattable1,
             crds1,
     board1,
     round1,
     currentbet - bet1[ind1],
     chips1[ind1],
     pot1,
     roundbets,
     blinds1[2],
     chips1,
     ind1,
     dealer1,
     ntable1))
    # cat("\n Seat ", ind1,": b1 was ",b1," and it was ",currentbet-bet1[ind1]," to him.")
    if(b1 > chips1[ind1]) b1 = chips1[ind1] ## if bet is more than you have, fix that.
    if(b1 > bmax1 - bet1[ind1]) b1 = bmax1 - bet1[ind1]
    ## can't bet more than anyone has left
     ## if bet is between 0.5 and the amount to you, make it a call.
     if((b1 > 0.5) && (b1 < currentbet - bet1[ind1])) b1 = min(chips1[ind1],
     currentbet - bet1[ind1])
```

```
## if bet is a raise of less than the big blind, make it a raise of the big blind.
raiseamt1 = b1 - (currentbet - bet1[ind1])
if((raiseamt1 > 0.5) && (raiseamt1 < blinds1[2])) b1 = min(blinds1[2] +
currentbet - bet1[ind1],chips1[ind1])
# cat("Now b1 = ",b1,". Currentbet = ",currentbet,".\n")
if(b1 > currentbet-bet1[ind1]+.5){ ## raise
    prevbet = currentbet
    currentbet = b1+bet1[ind1]
    # cat("curbet = ",currentbet,".\n")
    better1 = ind1
    pot1 = pot1 + b1
    roundbets[ind1,round1] = roundbets[ind1,round1] + b1
    bet1[ind1] = roundbets[ind1,round1]
    in1[ind1] = 1
    chips1[ind1] = chips1[ind1] - b1
} else if(b1 == min(chips1[ind1],currentbet-bet1[ind1])){ ## call
    pot1 = pot1 + b1
    roundbets[ind1,round1] = roundbets[ind1,round1] + b1
    bet1[ind1] = roundbets[ind1,round1]
    in1[ind1] = 1
    chips1[ind1] = chips1[ind1] - b1
} else if((chips1[ind1]>0.5) && (b1 < min(chips1[ind1],
currentbet-bet1[ind1])-.5)){ ## fold
    in1[ind1] = 0
    myout1 = 2
}
betlist1 = c(betlist1,bet1[ind1])
indlist1 = c(indlist1,ind1)
outlist1 = c(outlist1,myout1)
    }
    ind1 = ind1 + 1
    if(ind1 > numattable1) ind1 = 1
    if(better1 == ind1) stp = 2
    # cat("\n\n who:",numattable1,playerseats1, chips1,".\n")
    if(sum(in1) < 1.5) stp = 2
}
z3 = 0 ## now see if all the betting is over: if so, let z3 = 2.
if(sum(chips1[c(1:numattable1)[in1 > .5]] > .5) < 1.5) z3 = 2
# cat("\n Round 1 over.  z3 = ",z3,". in1 = ",in1,".\n")
list(i1=in1,p1=pot1,c1=chips1,rb=roundbets,all1=z3,bl1 = betlist1,
il1 = indlist1,out1=outlist1)
} ## end of bid1
```

---

bid2 *Postflop bidding*

Description

Runs a round of postflop bidding. Used by tourn1().

Usage

```
bid2(numattable1, playerseats1, blinds1, dealer1, b3, b4, round1, ntable1, decision1)
```

Arguments

numattable1
playerseats1
blinds1
dealer1
b3
b4
round1
ntable1
decision1

Details

Similar to bid1(), but starting with the player to the left of the dealer, and no small and big blinds.

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, playerseats1, blinds1, dealer1, b3, b4, round1, ntable1, decision1){
   if(b4$all1 > 1) return(b4)
   in1 = b4$i1
   if(sum(in1)<1.5) return(b4)
   betlist1 = c(0)
   indlist1 = c(0)
   outlist1 = c(0)
   chips1 = b4$c1
   bet1 = rep(0,numattable1)
   ind1 = dealer1+1
   if(ind1>numattable1) ind1 = 1
   currentbet = 0
   better1 = dealer1+1
   if(better1 > numattable1) better1 = 1
   v1 = c(0,3,4,5)[round1]
   board1 = matrix(rep(0,10),ncol=2)
   board1[1:v1,1] = b3$brdnum1[1:v1]
   board1[1:v1,2] = b3$brdsuit1[1:v1]
```

```
roundbets = b4$rb
pot1 = b4$p1
stp = 0
while(stp < 1){
    out1 = 0
    if(in1[ind1] > 0.5){
crds1 = matrix(c(b3$plnum1[ind1,],b3$plsuit1[ind1,]),ncol=2,byrow=F)
# cat("...",playerseats1[ind1],"'s turn...")
bmax1 = max((bet1[-ind1] + chips1[-ind1])[in1[-ind1]>.5])
b1 = round(decision1[[playerseats1[ind1]]](numattable1,
         crds1,
 board1,
 round1,
 currentbet - bet1[ind1],
 chips1[ind1],
 pot1,
 roundbets,
 blinds1[2],
 chips1,
 ind1,
 dealer1,
 ntable1))
# cat("\n Seat ", ind1,": b1 was ",b1," and it was ",currentbet-bet1[ind1]," to him.")
if(b1 > chips1[ind1]) b1 = chips1[ind1] ## if bet is more than you have, fix that.
if(b1 > bmax1 - bet1[ind1]) b1 = bmax1 - bet1[ind1]
## can't bet more than anyone else who's in has left
## if bet is between 0.5 and the amount to you, make it a call.
if((b1 > 0.5) && (b1 < currentbet - bet1[ind1])) b1 = min(chips1[ind1],
currentbet - bet1[ind1])
## if bet is a raise of less than the big blind, make it a raise of the big blind.
raiseamt1 = b1 - (currentbet - bet1[ind1])
if((raiseamt1 > 0.5) && (raiseamt1 < blinds1[2])) b1 = min(blinds1[2] +
currentbet - bet1[ind1],chips1[ind1])
if(b1 > currentbet - bet1[ind1]+.5){ ## raise
    if(b1 > chips1[ind1]) b1 = chips1[ind1]
    currentbet = b1 + bet1[ind1]
    better1 = ind1
    pot1 = pot1 + b1
    roundbets[ind1,round1] = roundbets[ind1,round1] + b1
    bet1[ind1] = roundbets[ind1,round1]
    in1[ind1] = 1
    chips1[ind1] = chips1[ind1] - b1
} else if(b1 == min(chips1[ind1],currentbet-bet1[ind1])){ ## call/check
    pot1 = pot1 + b1
    roundbets[ind1,round1] = roundbets[ind1,round1] + b1
    bet1[ind1] = roundbets[ind1,round1]
    in1[ind1] = 1
    chips1[ind1] = chips1[ind1] - b1
} else if((chips1[ind1]>0.5) && (b1 < min(chips1[ind1],
currentbet-bet1[ind1]))){ ## fold
    in1[ind1] = 0
    out1 = 2
}
```

```
    betlist1 = c(betlist1,bet1[ind1])
    indlist1 = c(indlist1,ind1)
    outlist1 = c(outlist1, out1)
        }
        ind1 = ind1 + 1
        if(ind1 > numattable1) ind1 = 1
        if(better1 == ind1) stp = 2
        if(sum(in1) < 1.5) stp = 2
    }
    z3 = 0 ## now see if all the betting is over: if so, let z3 = 2.
    if(sum(chips1[c(1:numattable1)[in1 > .5]] > .5) < 1.5) z3 = 2
    list(i1=in1,p1=pot1,c1=chips1,rb=roundbets,all1=z3,bl1 = betlist1,
    il1 = indlist1,out1=outlist1)
   } ## end of bid2
```

`calcwin1` *Figure out who won the hand.*

Description

Figures out who wins the hand. Used by tourn1().

Usage

```
calcwin1(numattable1, playerseats1, b3, b7)
```

Arguments

`numattable1`

`playerseats1`

`b3`

`b7`

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1,playerseats1, b3, b7){
## First, for everyone who's in, evaluate their hands.
    qual1 = rep(0,numattable1)
    inmuch1 = rep(0,numattable1)
    for(i in c(1:numattable1)){
```

```
if(b7$i1[i] > .5) {
    qual1[i] = handeval(c(b3$brdnum1[1:5],b3$plnum1[i,1:2]),
c(b3$brdsuit1[1:5],b3$plsuit1[i,1:2]))
}
inmuch1[i] = sum(b7$rb[i,])
    }
    z1 = order(unique(inmuch1[inmuch1 > 0.5]))
    difbets = sum(unique(inmuch1[inmuch1 > 0.5])>0) ## How many unique betting amounts.
    pot1 = b7$p1
    chips1 = b7$c1
    prevamount1 = 0
    for(j in c(1:difbets)){
amount1 = unique(inmuch1[inmuch1 > 0.5])[z1[j]] ## Start with the least amount
n1 = sum(inmuch1 >= amount1)  ## How many bet at least that much.
pl1 = c(1:numattable1)[inmuch1 >= amount1]  ## Who bet at least that much
# winner1 = pl1[order(qual1[pl1],decreasing=T)[1]]
winner1 = pl1[c(1:n1)[qual1[pl1]==max(qual1[pl1])]]
## Who has best hand (may be vector, if ties)
more1 = amount1 - prevamount1
for(k in winner1) chips1[k] = chips1[k] + round(n1*more1/length(winner1))
pot1 = pot1 - n1*more1
prevamount1 = amount1
    }
    chips1
  } ## end of calcwin1
```

`deal1` *Deals the cards.*

Description

Deals cards to the different players, and also deals the board to come. Used by tourn1().

Usage

```
deal1(numpl)
```

Arguments

`numpl`	number of players at the table.

Details

Each player's cards will be *ordered*, so that the number of your 1st card is at least as big as the number of your 2nd card.

Value

plnum1	vector of numbers (2=2, 3=3, ..., 13 = king, 14 = ace) of the players' cards.
plsuit1	vector of suits (1-4) of the players' cards.
brdnum1	vector of numbers (2-14) of the 5 board cards.
brdsuit1	vector of suits (1-4) of the board cards.

Author(s)

Frederic Paik Schoenberg

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
## out of 1000 hands with 2 players, see how many times
## both players get at least a straight, if neither ever folds.
n = 1000
result = rep(0,n)
for(i in 1:n){
x1 = deal1(2)
b1 = handeval(c(x1$plnum1[1,],x1$brdnum1), c(x1$plsuit1[1,],x1$brdsuit1))
b2 = handeval(c(x1$plnum1[2,],x1$brdnum1), c(x1$plsuit1[2,],x1$brdsuit1))
if(min(b1,b2) > 4000000) result[i] = 1
}
sum(result>.5)
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numpl){
    ## numpl is the number of players at the table
numcards = 2*numpl+5
crds1 = order(runif(52))[1:numcards]
crds2 = switch2(crds1)
num1 = crds2$num
suit1 = crds2$st
brdnum1 = num1[(numcards-4):numcards]
brdsuit1 = suit1[(numcards-4):numcards]
plnum1 = matrix(num1[1:(2*numpl)],ncol=2)
plsuit1 = matrix(suit1[1:(2*numpl)],ncol=2)
## order them
for(i in c(1:numpl)){
if(plnum1[i,1]<plnum1[i,2]){
a = plnum1[i,1]
plnum1[i,1] = plnum1[i,2]
plnum1[i,2] = a
```

```
a = plsuit1[i,1]
plsuit1[i,1] = plsuit1[i,2]
plsuit1[i,2] = a
  }
  }
b9 = list(plnum1=plnum1, plsuit1=plsuit1,
brdnum1=brdnum1, brdsuit1=brdsuit1)
b9
  } ## end of deal1
```

`flush1` *See if you have a flush*

Description

Sees if you have a flush, and if so, how high it is.

Usage

```
flush1(x, y)
```

Arguments

`x`	vector of numbers of your cards and the board cards, together.
`y`	corresponding suits of your cards and the board cards.

Details

Can break down if dealing with 10 cards or more, i.e. if 2 flushes are possible

Value

returns 0 if you don't have a flush, or 15^4 times top number + 15^3 times 2nd-highest + ... if you do.

Author(s)

Frederic Paik Schoenberg

Examples

```
x = c(2,14,8,8,5,3,4)
y = c(1,2,4,1,1,1,1)
flush1(x,y)
x = c(2,14,8,8,5,3,4)
y = c(1,2,4,1,3,1,1)
flush1(x,y)
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
```

```
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x,y){
    ## Can break down if dealing with 10 cards or more, i.e. if 2 flushes are possible
    ## returns 15^4 times top number + 15^3 times 2nd-highest + ...
a1 = mycount1(y)
if(max(a1$ct)<4.5) return(0)
a2 = c(1:length(a1$ct))[a1$ct > 4.5]
a3 = a1$v[a2] ## this is the suit
a4 = sort(x[y == a3],decreasing=T)
sum(15^c(4:0) * a4[1:5])
  } ## end of flush1
```

`flushdraw1` *Sees if you have a flush draw.*

Description

Calculates how many you have of the suit of which you have most.

Usage

```
flushdraw1(x)
```

Arguments

x	Suits of your cards and the board cards (together, as a vector).

Value

returns the number you have of a suit, i.e. 4 if you have a flush draw, or 5 (or more) if you have a flush already.

Author(s)

Frederic Paik Schoenberg

Examples

```
x = c(1,2,4,1,3,1,1)
flushdraw1(x)
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x){
    a1 = mycount1(x)
    max(a1$ct)
  } ## end of flushdraw1
```

four1 *Sees if you have 4 of a kind*

Description

Sees if you have 4 of a kind.

Usage

```
four1(x)
```

Arguments

x	Numbers of your cards and the board cards.

Value

15*number of the foursome + the number of your next highest other card. For instance, if your best 5 card hand is 8888K, then it will return 15*8 + 13 = 133. Returns 0 if you don't have 4 of a kind.

Author(s)

Frederic Paik Schoenberg

Examples

```
x = c(8,8,8,8,13,12,2)
four1(x)
x = c(8,8,8,3,13,12,2)
four1(x)
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x){
## 15*number of the foursome + next
a1 = mycount1(x)
a2 = a1$v
a3 = a1$ct
a4 = sum(a3 > 3.5)
if(a4 < 0.5) return(0)
a5 = sort(a2[a3>3.5],decreasing=T)
a6 = sort(c(0,x[(x != a5[1])]),decreasing=T)
15*a5[1] + a6[1]
  }    ## end of four1
```

full1 *See if you have a full house.*

Description

Sees if you have a full house.

Usage

```
full1(x)
```

Arguments

x	Numbers of your cards and the board cards.

Value

15 times the number of your triplet + the number of your pair in your best five card hand, or 0 if you don't have a full house. For instance, if you have 22 and the board is 2AAQQ, then it returns 15*2 + 14 = 44.

Examples

```
x = c(2,2,2,14,14,12,12)
full1(x)
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x){
a1 = mycount1(x)
a2 = sort(a1$ct,decreasing=T)
if(length(a2) < 1.5) return(0)
if(a2[1] < 2.5) return(0)
if(a2[2] < 1.5) return(0)
a3 = min(c(1:length(a1$ct))[a1$ct > 2.5])
a4 = a1$v[a3] ## the number of the trip
a5 = a1$ct[-a3]
a6 = a1$v[-a3]
a7 = min(c(1:length(a5))[a5 > 1.5]) ## the number of the pair
a8 = a6[a7]
15*a4 + a8
  }  ## end of full1
```

gravity *A sample function that goes all in or folds*

Description

Goes all in with any pair of 10s or greater, or AJ-AK, or suited connectors with lowest card a 10 or higher, or, if your chip count is less than twice the big blind, then it goes all in with any 2 cards.

Usage

```
gravity(numattable1, crds1, board1, round1, currentbet, mychips1,
pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft)
```

Arguments

numattable1	(integer) number of players at the table
crds1	(2x2 matrix) your hole cards. cards1[1,1] = the number of card 1 (between 2 and 14). cards[1,2] = suit of card 1 (between 1 and 4). cards1[2,1] = the number (2-14) of card 2. cards[2,2] = suit of card 2 (1-4).
board1	(5x2 matrix) the board cards. board1[1,1] = number of first card (2-14). board1[1,2] = suit (1-4) of first card. Both are zero if the card hasn't been seen yet. For instance, if bettinground1 < 3, then board[4,1] = board1[4,2] = 0.
round1	(integer) which betting round it is. 1 = preflop, 2 = after flop, 3 = after turn, 4 = after river.
currentbet	(integer) how much more it is to you to stay in, right now.
mychips1	(integer) how many chips you have left at the moment.
pot1	(integer) how much is in the pot at the moment.
roundbets	(numattable1 x 4 matrix) matrix of all past bets during this hand. roundbets[i,j] = total amount the player in seat i put in, in betting round j.
blinds1	(integer) big blind amount.
chips1	(vector of length numattable1) list of how many chips everyone has. chips1[i] = how many chips the player in seat i has left.
ind1	(integer) which seat you're in. (So, mychips1 = chips1[ind1]).
dealer1	(integer) which seat the dealer is in.
tablesleft	(integer) how many tables are left in the tournament (including yours).

Details

When this function is called from tourn1(), cards1[2,1] is always less than or equal to cards1[1,1]. If the player in the big blind seat does not have enough chips to pay the big blind, then blinds1 is still the amount that the big blind would have been. If only 2 players are left, then tourn1() uses the convention that the "dealer" is the big blind in determining dealer1.

Value

integer indicating the number of chips you are betting. 0 means fold.

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, crds1, board1,  round1, currentbet,
    mychips1, pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft){
    ## all in with any pair of 10s or greater,
    ## or AJ-AK, or suited connectors with lowest card a 10 or higher.
    ## if your chip count is less than twice the big blind, go all in
    ## with any cards.
    a1 = 0
    if((crds1[1,1] == crds1[2,1]) && (crds1[1,1] > 9.5)) a1 = mychips1
    if((crds1[1,1] > 13.5) && (crds1[2,1]>10.5)) a1 = mychips1
    if((crds1[1,1]-crds1[2,1]==1) && (crds1[1,2] == crds1[2,2]) &&
(crds1[2,1]>9.5)) a1 = mychips1
    if(mychips1 < 2*blinds1)  a1 = mychips1
    a1
  } ## end of gravity
```

---

hand1 *Play a hand of Texas Hold'em*

Description

Used in tourn1() to play a single hand of Texas Hold'em.

Usage

```
hand1(numattable1, playerseats1, chips1, blinds1, dealer1, ntable1,
myfast1, t1, t2, chipstart1, lowercut1, decision1,
name1 = c("vera","william","xena","yosef","zelda"))
```

Arguments

numattable1	number of players at the table
playerseats1	list of indices, who's in seat 1, seat 2, etc.
chips1	list of chips left, for players at this table only.

blinds1	vector of small and big blind
dealer1	seat that the dealer is in
ntable1	how many tables remain in the tournament
myfast1	2 = fast, or 0 = slow, i.e. 2 means it will not show the hand graphically.
t1	probability of showing the hand graphically if there is a double up
t2	probability of showing the hand graphically if there is an elimination
chipstart1	starting number of chips everyone has
lowercut1	lower plotting threshold for a player's number of chips (the y axis)
decision1	a vector of the players' codes
name1	names of the players

Value

chips2	how many chips everyone has after the hand
draw1	2 if the hand is to be shown graphically, or 0 if not

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, playerseats1, chips1, blinds1, dealer1,
ntable1,myfast1,t1,t2,chipstart1,lowercut1, decision1){
    ## numattable1 = number of players at the table
    ## playerseats1 = list of indices, who's in seat 1, seat 2, etc.
    ## chips1 = list of chips left, FOR PLAYERS AT THIS TABLE ONLY!
    ## blinds = vector of small and then big blind
    ## dealer1 = seat that the dealer is in.
    ## ntable1 = how many tables remain.
    chips2 = chips1 ## this will be the revised chips counts, at the end
    if(numattable1 < 1.5) return(chips2)
          ## want to return other stuff too, liks who's left?
          ## No need... can do that within main loop. Just check for zeros.
    b3 = deal1(numattable1)
    b4 = bid1(numattable1,playerseats1, chips1, blinds1, dealer1, b3, ntable1, decision1)
    # cat("\n...",b4$bl1,"\n....",b4$il1,"\n")
    b5 = bid2(numattable1,playerseats1, blinds1, dealer1, b3,b4,2, ntable1, decision1)
    b6 = bid2(numattable1,playerseats1, blinds1, dealer1, b3,b5,3, ntable1, decision1)
    b7 = bid2(numattable1,playerseats1, blinds1, dealer1, b3,b6,4, ntable1, decision1)
    chips2 = calcwin1(numattable1,playerseats1, b3, b7)
    draw1 = 0
    u21 = runif(1)
```

```
    if(((max(chips2/(chips1+.01)) > 1.99) &&
    (u21 < t1)) || ((max(chips1/(chips2+.01)) > 99) && (u21 < t2))) {
if(myfast1 < 1) {
    text(1,lowercut1,"click to continue",cex=.7)
    locator(1)
}
mygraphics1(numattable1,playerseats1,chips1,blinds1,dealer1,
b3,b4,b5,b6,b7,chips2,ntable1,myfast1,chipstart1,
    name1,lowercut1)
draw1 = 2
    }
    list(chips2=chips2,draw1=draw1)
  }  ## end of hand1
```

---

handeval *Evaluates your hand.*

Description

Returns a number corresponding to the strength of your best 5-card poker hand.

Usage

```
handeval(num1, suit1)
```

Arguments

num1	vector of numbers of your cards and the board cards.
suit1	corresponding suits of your cards and the board cards.

Value

If you have a straight flush, then it will return a value in the 8 million to 9 million range, if you have four of a kind, then it will return a value between 7 million and 8 million, etc., according to the list below.

Straight Flush: 8,000,000 - 8,999,999 Four of a kind: 7,000,000 - 7,999,999. Full House: 6,000,000 - 6,999,999. Flush: 5,000,000 - 5,999,999. Straight: 4,000,000 - 4,999,999. 3 of a kind: 3,000,000 - 3,999,999. Two pairs: 2,000,000 - 2,999,999. One pair: 1,000,000 - 1,999,999. No pairs: 0 - 999,999. If one player's hand beats that of another, then the value returned by this function will be higher for the first player than the second.

Author(s)

Frederic Paik Schoenberg

Examples

```
boardcards = c(4,5,6,8,13)
boardsuits = c(2,3,2,2,2)
player1cards = c(2,3)
player1suits = c(2,1)
player2cards = c(7,3)
player2suits = c(2,4)
handeval(c(boardcards,player1cards),c(boardsuits,player1suits)) ## pl.1's value
handeval(c(boardcards,player2cards),c(boardsuits,player2suits)) ## pl.2's value

##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(num1,suit1){
    a1 = strflsh1(num1,suit1)
    if(a1>0.5) return(8000000+a1)
    a1 = four1(num1)
    if(a1>0.5) return(7000000+a1)
    a1 = full1(num1)
    if(a1>0.5) return(6000000+a1)
    a1 = flush1(num1,suit1)
    if(a1>0.5) return(5000000+a1)
    a1 = straight1(num1)
    if(a1>0.5) return(4000000+a1)
    a1 = trip1(num1)
    if(a1>0.5) return(3000000+a1)
    a1 = twopair1(num1)
    if(a1>0.5) return(2000000+a1)
    a1 = onepair1(num1)
    if(a1>0.5) return(1000000+a1)
    a1 = nothing1(num1)
    return(a1)
  } ## end of handeval
```

---

many1 *function to run many Texas Hold'em tournaments.*

Description

Runs the desired number of Texas Hold'em tournaments, and saves the results. By default, only the top 3 finishers receive payouts, and their payouts are 13, 8, and 5, respectively.

Usage

```
many1(name1, decision1, k, winners1 = 3, payouts1 = c(13, 8, 5))
```

Arguments

name1	names of the players
decision1	codes governing the players' betting
k	number of players
winners1	number of top finishers receiving payouts in each tournament
payouts1	payouts for the top finishers

Details

If length(name1) < k or length(decision1) < k, there will be errors.

Author(s)

Frederic Paik Schoenberg

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
name1 = c("gravity","tommy","ursula","timemachine","vera","william","xena")
decision1 = list(gravity, tommy, ursula, timemachine, vera, william, xena)
z1 = many1(name1, decision1, 7)
z2 = z1[[2]]
barplot(z2,names.arg=name1[1:7],cex.names=.9)

## The function is currently defined as
function(name1, decision1, k,winners1=3, payouts1 = c(13,8,5)){
    ## runs k tournaments
    nplayers1 = length(name1)
    d1 = matrix(0,ncol=winners1,nrow=k) ## matrix of results
    d2 = rep(0,nplayers1) ## total pts for each player
    for(i in 1:k){
cat("\n\n.....  TOURNAMENT NUMBER ",i,":  ..... \n\n")
d1[i,] = tourn1(name1, decision1)
for(j in 1:winners1) d2[d1[i,j]] = d2[d1[i,j]] + payouts1[j]
print(rbind(1:nplayers1,d2))
#locator(1)
    }
    list(d1,d2)
  } ## end of many1
```

mycount1 *counts the unique values in a vector*

Description

returns sorted unique values of x and how many times each appears

Usage

```
mycount1(x)
```

Arguments

x

Value

and their cardinality

v	sorted unique values in x
ct	cardinality of each of the values

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x){
    ## returns sorted unique values of x and how many times each appears
b1 = sort(unique(x),decreasing=T)
b2 = length(b1)
b3 = rep(0,b2)
for(i in 1:b2) b3[i] = sum(x == b1[i])
list(v=b1, ct = b3)
  }
```

mygraphics1 *Graphically show a poker hand*

Description

On the plotting device, show a poker hand. Used by tourn1() and hand1().

Usage

```
mygraphics1(numattable1, playerseats1, chips1, blinds1,
dealer1, b3, b4, b5, b6, b7, chips2, ntable1, myfast1, chipstart1, name1, lowercut1)
```

Arguments

numattable1

playerseats1

chips1

blinds1

dealer1

b3

b4

b5

b6

b7

chips2

ntable1

myfast1

chipstart1

name1

lowercut1

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, playerseats1, chips1, blinds1, dealer1,b3,b4,b5,b6,b7,chips2,
```

```
       ntable1,myfast1,chipstart1,name1,lowercut1){
    nplayers1 = length(name1)
    cardname1 = c(as.character(1:9),"T","J","Q","K","A")
    suitname1 = c(2,3,4,6)
    plot(c(0,10+10*numattable1),c(0,100),type="n",xlab="",ylab="",xaxt="n",yaxt="n")
    a1 = paste("Blinds are",blinds1[1],"and",blinds1[2],".")
    text(10,95,a1)
    if(ntable1>1.5) a2 = paste(ntable1,"tables left.") else a2 = "1 table left."
    text(10*numattable1,95,a2)
    text(10*dealer1,90,"D")
    text(5,5,"click to continue",cex=.8)
    for(j in c(1:numattable1)){
i = 2 + dealer1 + j
if(i > numattable1 + .5) i = i - numattable1
if(i > numattable1 + .5) i = i - numattable1
text(10*i,80,as.character(name1[playerseats1[i]]),cex=1+.1*b7$i1[i],col=1)
a1 = paste("(",chips1[i],")")
text(10*i,75,a1)
text(10*i,50,"BETS:")
    }
    ## pre-flop bets:
    writebets1(b4,1,numattable1,b3,playerseats1,chips1,chips2,0,myfast1,name1)
## Now flop:
    for(j in c(1:3)){
text(j/6*(10+10*numattable1),15,
cardname1[b3$brdnum1[j]],col=suitname1[b3$brdsuit1[j]],cex=2)
    }
    if(myfast1<1) locator(1)
    over1 = b4$all1
    writebets1(b5,2,numattable1,b3,playerseats1,chips1,chips2,over1,myfast1,name1)
    ## Now turn:
    over1 = b5$all1
    j = 4
   text(j/6*(10+10*numattable1),15,cardname1[b3$brdnum1[j]],col=suitname1[b3$brdsuit1[j]],cex=2)
    if(myfast1<1) locator(1)
    writebets1(b6,3,numattable1,b3,playerseats1,chips1,chips2,over1,myfast1,name1)
    ## Now river:
    over1 = b6$all1
    j = 5
   text(j/6*(10+10*numattable1),15,cardname1[b3$brdnum1[j]],col=suitname1[b3$brdsuit1[j]],cex=2)
    if(myfast1<1) locator(1)
    writebets1(b7,4,numattable1,b3,playerseats1,chips1,chips2,over1,myfast1,name1)
    for(i in c(1:numattable1)) if(chips1[i] != chips2[i]) text(10*i, 70, chips2[i])
    for(i in c(1:numattable1)){
if(chips2[i]>chips1[i]){
    text(10*i,85,"$",cex=3)
    text(10*i,80,as.character(name1[playerseats1[i]]),cex=1+.1*b7$i1[i],col=5)
}
if(chips2[i] < .5){
    text(10*i,80,as.character(name1[playerseats1[i]]),cex=1+.1*b7$i1[i],col=2)
}
    }
    if(myfast1<1) locator(1)
```

```
  plot(c(0,nplayers1+1),c(lowercut1,chipstart1*nplayers1),
  type="n",xlab="player number",ylab="chips",log="y")
} ## end of mygraphics1
```

---

nothing1 *indicates strength of your hand when you have nothing*

Description

finds the strength of your hand when you have nothing. Used by handeval().

Usage

```
nothing1(x)
```

Arguments

x

Value

Returns 15^4*highest card + 15^3*next highest + 225*next + 15*next + next

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x){
## 15^4*highest + 15^3*next + 225*next + 15*next + next
y = c(x,rep(0,5))  ## this is in case x has length < 5
a1 = sort(y,decreasing=T)
15*15*15*15*a1[1] + 15*15*15*a1[2] + 225*a1[3] + 15*a1[4] + a1[5]
  } ## end of nothing1
```

onepair1 *See if you have a pair.*

Description

Sees if you have a pair, and if so, finds the strength of your hand. Used by handeval().

Usage

```
onepair1(x)
```

Arguments

x

Value

returns 15*15*15*pair + 15*15*next + 15*next + next if you have a pair, or 0 otherwise.

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x){
## 15*15*15*pair + 15*15*next + 15*next + next
a1 = unique(x)
a2 = length(a1)
if(a2 == length(x)) return(0)
a3 = rep(0,a2)
for(i in 1:a2) a3[i] = sum(x == a1[i])
a4 = max(a1[a3>1.5])
a5 = sort(c(x[x != a4],rep(0,3)),decreasing=T)
15*15*15*a4 + 225*a5[1] + 15*a5[2] + a5[3]
  }
```

straight1 *See if you have a straight.*

Description

Sees if you have a straight, and if so, finds the strength of your hand. Used by handeval().

Usage

```
straight1(x)
```

Arguments

x

Value

returns the highest card of your straight, or 0 if you don't have one.

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x){
a1 = sort(unique(x))
if (length(a1)<4.5) return(0)
a3 = 0
n = length(a1)
if(a1[n] == 14) a1 = c(1,a1) ## count ace as both 1 and 14
a2 = length(a1)
for(j in c(5:a2)){ ## j will be the potential highest card of straight
    if( sum(15^c(1:5) * a1[(j-4):j]) == sum(15^c(1:5) * ((a1[j]-4):a1[j]))) a3 = a1[j]
  }
a3
  } ## end of straight1
```

`straightdraw1` *See if you have a straight draw.*

Description

Sees how many possibilities there are that will make a straight for you.

Usage

```
straightdraw1(x)
```

Arguments

`x`

Value

returns 26 if you already have a straight. returns 4 is there are 2 possibilities for a straight. returns 2 for a gutshot straight draw. returns 0 otherwise.

Author(s)

Frederic Paik Schoenberg

Examples

```
straightdraw1(c(2,5,6,7,9,14))
straightdraw1(c(4,5,6,7,9,14))
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x){
    ## returns 4 is there are 2 possibilities for a straight.
    ## returns 2 for a gutshot straight draw.
    ## returns 0 otherwise
    ## Note: returns 26 if you already have a straight!
    a1 = 0
    for(i in c(2:14)){
a2 = straight1(c(i,x))
if(a2 > .5) a1 = a1 + 1
    }
    a1 * 2
  } ## end of straightdraw1
```

strflsh1 *See if you have a straight flush.*

Description

Sees if you have a straight flush, and if so, finds the strength of your hand. Used by handeval().

Usage

```
strflsh1(x, y)
```

Arguments

x

y

Value

returns the highest card of your straight flush if you have one, or 0 otherwise.

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x,y){
a1 = mycount1(y)
if(max(a1$ct)<4.5) return(0)
a2 = c(1:length(a1$ct))[a1$ct > 4.5]
a3 = a1$v[a2] ## this is the suit
a4 = sort(x[y == a3],decreasing=T)
straight1(a4)
  } ## end of strflush1
```

switch2 *Finds the cards corresponding to numbers between 1 and 52.*

Description

Takes each integer between 1 and 52 and turns it into a card. Used in deal1() and tourn1().

Usage

```
switch2(x)
```

Arguments

x	a vector of integers between 1 and 52.

Value

num	numbers (2-14) of the cards
st	suits (1-4) of the cards

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x){
    ## takes a number 1-52, and turns it into a card:
    ## returns a list, where the 1st is the number (2-14), and 2nd is suit (1-4).
    n = length(x)
    y = list(num=x, st=rep(1,n))
    for(i in c(1:n)){
a = 1
while(a>0){
    if(y$num[i]<14) a = -1 else{y$st[i] = y$st[i]+1
y$num[i] = y$num[i]-13
    }
}
    }
    y$num = y$num+1
    y
  } ## end of switch2
```

timemachine *A sample function that goes all in or folds*

Description

Goes all in with any pair 7 or higher, or AK, AQ, or AJ if nobody's all in yet, or with prob 75% if you have less than 3 big blinds and one card is 10 or higher.

Usage

```
timemachine(numattable1, crds1, board1, round1, currentbet, mychips1,
pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft)
```

Arguments

numattable1	(integer) number of players at the table
crds1	(2x2 matrix) your hole cards. cards1[1,1] = the number of card 1 (between 2 and 14). cards[1,2] = suit of card 1 (between 1 and 4). cards1[2,1] = the number (2-14) of card 2. cards[2,2] = suit of card 2 (1-4).
board1	(5x2 matrix) the board cards. board1[1,1] = number of first card (2-14). board1[1,2] = suit (1-4) of first card. Both are zero if the card hasn't been seen yet. For instance, if bettinground1 < 3, then board[4,1] = board1[4,2] = 0.
round1	(integer) which betting round it is. 1 = preflop, 2 = after flop, 3 = after turn, 4 = after river.
currentbet	(integer) how much more it is to you to stay in, right now.
mychips1	(integer) how many chips you have left at the moment.
pot1	(integer) how much is in the pot at the moment.
roundbets	(numattable1 x 4 matrix) matrix of all past bets during this hand. roundbets[i,j] = total amount the player in seat i put in, in betting round j.
blinds1	(integer) big blind amount.
chips1	(vector of length numattable1) list of how many chips everyone has. chips1[i] = how many chips the player in seat i has left.
ind1	(integer) which seat you're in. (So, mychips1 = chips1[ind1]).
dealer1	(integer) which seat the dealer is in.
tablesleft	(integer) how many tables are left in the tournament (including yours).

Details

When this function is called from tourn1(), cards1[2,1] is always less than or equal to cards1[1,1]. If the player in the big blind seat does not have enough chips to pay the big blind, then blinds1 is still the amount that the big blind would have been. If only 2 players are left, then tourn1() uses the convention that the "dealer" is the big blind in determining dealer1.

Value

integer indicating the number of chips you are betting. 0 means fold.

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, crds1, board1,  round1, currentbet,
    mychips1, pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft){
    ## any pair 7 or higher
    ## AK, AQ
    ## AJ if nobody's all in yet.
    ## if less than 3 times bb & one card is Ten or higher, then 75%
    a1 = 0
    x = runif(1)
    if((crds1[1,1] == crds1[2,1]) && (crds1[1,1] > 6.5)) a1 = mychips1
    if((crds1[1,1] == 14) && (crds1[2,1] > 11.5)) a1 = mychips1
    if((crds1[1,1] == 14) && (crds1[2,1] == 11) &&
(currentbet <= blinds1)) a1 = mychips1
    if((mychips1 < 3*blinds1) && (crds1[1,1] >= 10) && (x<.75)) a1 = mychips1
    a1
  } ## end of timemachine
```

---

tommy *A sample function that goes all in or folds*

Description

Goes all in with any pocket pair

Usage

```
tommy(numattable1, crds1, board1, round1, currentbet, mychips1,
pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft)
```

Arguments

numattable1	(integer) number of players at the table
crds1	(2x2 matrix) your hole cards. cards1[1,1] = the number of card 1 (between 2 and 14). cards[1,2] = suit of card 1 (between 1 and 4). cards1[2,1] = the number (2-14) of card 2. cards[2,2] = suit of card 2 (1-4).

board1	(5x2 matrix) the board cards. board1[1,1] = number of first card (2-14). board1[1,2] = suit (1-4) of first card. Both are zero if the card hasn't been seen yet. For instance, if bettinground1 < 3, then board[4,1] = board1[4,2] = 0.
round1	(integer) which betting round it is. 1 = preflop, 2 = after flop, 3 = after turn, 4 = after river.
currentbet	(integer) how much more it is to you to stay in, right now.
mychips1	(integer) how many chips you have left at the moment.
pot1	(integer) how much is in the pot at the moment.
roundbets	(numattable1 x 4 matrix) matrix of all past bets during this hand. roundbets[i,j] = total amount the player in seat i put in, in betting round j.
blinds1	(integer) big blind amount.
chips1	(vector of length numattable1) list of how many chips everyone has. chips1[i] = how many chips the player in seat i has left.
ind1	(integer) which seat you're in. (So, mychips1 = chips1[ind1]).
dealer1	(integer) which seat the dealer is in.
tablesleft	(integer) how many tables are left in the tournament (including yours).

Details

When this function is called from tourn1(), cards1[2,1] is always less than or equal to cards1[1,1]. If the player in the big blind seat does not have enough chips to pay the big blind, then blinds1 is still the amount that the big blind would have been. If only 2 players are left, then tourn1() uses the convention that the "dealer" is the big blind in determining dealer1.

Value

integer indicating the number of chips you are betting. 0 means fold.

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, crds1, board1,  round1, currentbet, mychips1,
    pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft){
    ## all in with any pair.
    a1 = 0
    if(crds1[1,1] == crds1[2,1]) a1 = mychips1
    a1
  } ## end of tommy
```

tourn1 *Run a Texas Hold'em tournament*

Description

Runs a Texas Hold'em tournament, stopping for certain key hands which are shown graphically. The top finishers are given payouts. By default, only the top 3 finishers get points, and they get 13, 8, and 5 points respectively. See chapter 8 of Schoenberg, F. (2011).

Usage

```
tourn1(name1, decision1, chipstart1 = 1000,
bigstart1 = 20, inc1 = 1.5, winners1 = 3, myfast1 = 2,
t1 = 0.5, t2 = 1, graphiccutoff1 = 0.1, lowercut1 = 30)
```

Arguments

name1	names of the players
decision1	the players' codes to determine their betting and folding
chipstart1	how many chips each player starts with
bigstart1	the initial big blind
inc1	how much the blinds increase every 10 hands. If inc1 = 2, for instance, then blinds double after every 10 hands.
winners1	number of top finishers receiving payouts.
myfast1	make it 2 if you want the tournament to run quickly, or 0 if you want to show graphics and have to click the mouse to proceed while showing each key hand.
t1	fraction of times to show double ups
t2	fraction of times to show eliminations
graphiccutoff1	If a student's chip stack changes by a factor less than this amount, then the graphic display will not bother replotting her name.
lowercut1	If a student has fewer chips than this, her name won't appear on the graphic display.

Details

Each tournament works as follows. Blinds last 10 hands, then increase by factor of inc1. (rounded to the nearest chip) If the number of players left is 11-20, then each hand, players are split into 2 tables of equal size (or one off if the number of players is odd). Then 10 hands are played, regardless of how many players are eliminated during those 10 hands. The small blind is always 1/2 the big blind (rounded to the nearest integer). If the number of players > 20, then each 10 hands, players are randomly split into tables of size 10. The remaining players not assigned to tables sit out these 10 hands.

t1 and t2 govern how many hands are "key" hands. Usually, every student wants to see at least 1 hand involving their code, so by default t2 = 1 so that each student's elimination is shown, and if t1 = 0.5, then if a student doubles up on a hand, then that hand is shown with probability 0.5. The code might have errors if winners1 > 10.

Value

A vector indicating the indices of the players who won, in order. For instance, an output of (4,1,5) means that function 4 got 1st place, function 1 got 2nd place, and function 5 got 3rd place.

Author(s)

Frederic Paik Schoenberg

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
name1 = c("gravity","tommy","ursula","timemachine","vera","william","xena")
decision1 = list(gravity, tommy, ursula, timemachine, vera, william, xena)
tourn1(name1, decision1, myfast1 = 2) ## run quickly
tourn1(name1, decision1, myfast1 = 0) ## run slowly, showing key hands

## The function is currently defined as
function(name1, decision1, chipstart1 = 1000, bigstart1 = 20,
    inc1 = 1.50, winners1 = 3,myfast1 = 2, t1 = 0.5, t2=1,
  graphiccutoff1 = .1, lowercut1 = 30 ){
    ## Note: don't have more than 10 winners! That might mess this up.
    # blinds last 10 hands, then increase by factor of inc1. (rounded)
    # If # players left is 11-20, then each hand, players are split
   # into 2 tables of equal size (or one off if # players is odd). Then 10 hands are played,
    # no matter what.
    # Small = 1/2 big (rounded to nearest integer).
    # If num players > 20, then each 10 hands, players are randomly split
    # into tables of size 10. Remaining players sit out 10 hands.
    # a) Start loop. Initiate blinds.
    # b) Assign seats. Set num of tables.
    # c) For each table, play a hand. Repeat 10 times.
    # d) After each hand, update nplayers1, and
    # if nplayers1 <= winners1, then store winners. See if all done too.
    # e) On key hands, do instant replay! (if all-in & call, and
    #    total pool > 8 times big.
    # f) Increase blinds and repeat b-e.

    ## myfast1 = 2 if you want it to go fast. 0 = slow.
    ## t1 = fraction of times to show double-ups
    ## t2 = fraction of times to show eliminations
    ## chipstart1 = starting number of chips. The big blind starts at 20.
    ## winners1 = the number of function that get points.
    ## payouts1 = c(13,8,5) = the number of points for the winner, 2nd place, etc.

    nplayers1 = length(name1)
    plot(c(0,nplayers1+1),c(lowercut1,chipstart1*nplayers1),pch=name1[1:nplayers1],
type="n",xlab="player number",ylab="chips",log="y")
    chip1 = rep(chipstart1, nplayers1)
```

```
    text(x=c(1:nplayers1),y=chip1,cex=2,labels=name1[1:nplayers1],srt=270,col=2)
    chip7 = chip1
    big1 = round(bigstart1)
    sm1 = round(big1/2)
    blinds1 = c(sm1,big1)
    nleft1 = nplayers1
    plleft = 1:nplayers1   ## plleft will be the indices of who's left.
    places1 = rep(0,winners1)

    stp3 = 0
    while(stp3 < 1){
if(nleft1 > 20.5){
    ntable1 = floor(nleft1/10)
    pl1 = sample(nleft1)
    tables1 = list(tbnums = rep(10,ntable1))
    for(j in c(1:ntable1)){
tables1[[1+j]] = plleft[pl1[(1:10)+(j-1)*10]]
    }
}
if((nleft1 < 20.5) && (nleft1 > 10.5)){
    ntable1 = 2
    pl1 = sample(nleft1)
    thalf1 = ceiling(nleft1/2)
    bhalf1 = nleft1 - thalf1
    tables1 = list(tbnums = c(thalf1, bhalf1))
    tables1[[2]] = plleft[pl1[1:thalf1]]
    tables1[[3]] = plleft[pl1[(thalf1+1):nleft1]]
}
if(nleft1 < 10.5){
    ntable1 = 1
    pl1 = sample(nleft1)
    tables1 = list(tbnums = nleft1)
    tables1[[2]] = plleft[pl1]
    ## so, with 10 players or fewer, I'm re-shuffling seats every 10 hands.
}
cat("\n Big blind is ",blinds1[2],"\n")
for(i in 1:ntable1){
    k = 0
    for(j in 1:10){
chip3 = chip1[tables1[[1+i]]]
k = k+1
if(k > tables1[[1]][i]) k = 1
cat(j)
x32 = hand1(tables1[[1]][i], tables1[[1+i]], chip3,
blinds1, k, ntable1,myfast1,t1,t2,chipstart1,lowercut1, decision1)
chip2 = x32$chips2
chip1[tables1[[1+i]]] = chip2
chipdif8 = (abs(chip1-chip7)/pmax(chip1,chip7,rep(1,nplayers1)) > graphiccutoff1)
if(x32$draw1 > 1){
    text(x=c(1:nplayers1),y=chip1,cex=2,labels=name1[1:nplayers1],srt=270,col=2)
    chip7 = chip1
} else if(sum(chipdif8)>.5){
    text(x=c(1:nplayers1)[chipdif8],
```

```
    y=chip7[chipdif8],cex=2,col="white",labels=name1[chipdif8],srt=270)
    text(x=c(1:nplayers1)[chipdif8],
    y=chip7[chipdif8],cex=2,col="white",labels=name1[chipdif8],srt=270)
    text(x=c(1:nplayers1)[chipdif8],
    y=chip1[chipdif8],cex=2,col=2,labels=name1[chipdif8],srt=270)
    chip7[chipdif8] = chip1[chipdif8]
}
## Now remove eliminated players, even if they were blinds.
## This may let some people miss their big blind. Note this.
j1 = sum(chip2 < .5) ## the number eliminated.
if(j1 > .5){
    j2 = tables1[[i+1]][c(1:tables1[[1]][i])[chip2 < .5]] ## their indices
    j3 = j2[order(chip3[j2],decreasing=T)] ## ordered by how much they had before
    j4 = min(winners1,nleft1)
    nleft1 = nleft1 - j1
    if(nleft1 < winners1 - .5) places1[(nleft1+1):j4] = j3[1:(j4-nleft1)]
    cat("\n Eliminated: ",j2,".....",nleft1," players remaining.\n")
    tables1[[1]][i] = tables1[[1]][i] - j1
    tables1[[1+i]] = tables1[[1+i]][chip2>.5]
}
if(nleft1 < 1.5) break
    }
    if(nleft1 < 1.5) break
}
big1 = round(blinds1[2]*inc1)
sm1 = round(big1/2)
blinds1 = c(sm1,big1)
plleft = c(1:nplayers1)[chip1>0.5]
nleft1 = length(plleft)
if(nleft1 < 1.5){
    stp3 = 2
    places1[1] = plleft
    z2 = winners1+1
   plot(c(0,nplayers1+1),c(lowercut1,chipstart1*nplayers1),
       type="n",xlab="player number",ylab="chips",log="y")
   text(1*nplayers1/z2,nplayers1*chipstart1, "1st:",col=4,cex=2)
    text(2*nplayers1/z2,nplayers1*chipstart1, "2nd:",col=4,cex=2)
    text(3*nplayers1/z2,nplayers1*chipstart1, "3rd:",col=4,cex=2)
    if(z2 > 4.5) for(z1 in c(4:winners1))
    text(z1*nplayers1/z2,nplayers1*chipstart1, paste(z1,"th:"),col=4,cex=2)
    for(z1 in c(1:winners1)) text(z1*nplayers1/(winners1+1),
    (nplayers1/2)*chipstart1,name1[places1[z1]],col=4,cex=2)
}
    }
    places1
  } ## end of tourn1
```

---

trip1 *See if you have 3 of a kind.*

Description

Sees if you have 3 of a kind, and if so, finds the strength of your hand. Used by handeval().

Usage

```
trip1(x)
```

Arguments

```
x
```

Value

Returns 225 * your triple + 15 * your next highest card + your next highest card if you have 3 of a kind, or 0 otherwise.

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x){
## 225*triple + 15*next + next
a1 = mycount1(x)
a2 = a1$v
a3 = a1$ct
a4 = sum(a3 > 2.5)
if(a4 < 0.5) return(0)
a5 = sort(a2[a3>2.5],decreasing=T)
a6 = sort(c(0,0,x[(x != a5[1])]),decreasing=T)
225*a5[1] + 15*a6[1] + a6[2]
  }    ## end of trip1
```

---

twopair1 *See if you have two pairs.*

Description

Sees if you have two pairs, and if so, finds the strength of your hand. Used by handeval().

Usage

```
twopair1(x)
```

Arguments

x

Value

225 * your higher pair + 15 * the lower pair + the number of your kicker, if you have two pairs, or 0 otherwise.

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(x){
## 225*highpair + 15*lowpair + next
a1 = mycount1(x)
a2 = a1$v
a3 = a1$ct
a4 = sum(a3>1.5)
if(a4<1.5) return(0)
a5 = sort(a2[a3>1.5],decreasing=T)
a6 = max(c(0,a2[(a2 != a5[1]) & (a2 != a5[2])]))
225*a5[1] + 15*a5[2] + a6
  }    ## end of twopair1
```

---

ursula *A sample function that goes all in or folds*

Description

Goes all in with a pocket pair of 9s or better.

Usage

```
ursula(numattable1, crds1, board1, round1, currentbet, mychips1,
pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft)
```

Arguments

numattable1	(integer) number of players at the table
crds1	(2x2 matrix) your hole cards. cards1[1,1] = the number of card 1 (between 2 and 14). cards[1,2] = suit of card 1 (between 1 and 4). cards1[2,1] = the number (2-14) of card 2. cards[2,2] = suit of card 2 (1-4).
board1	(5x2 matrix) the board cards. board1[1,1] = number of first card (2-14). board1[1,2] = suit (1-4) of first card. Both are zero if the card hasn't been seen yet. For instance, if bettinground1 < 3, then board[4,1] = board1[4,2] = 0.
round1	(integer) which betting round it is. 1 = preflop, 2 = after flop, 3 = after turn, 4 = after river.
currentbet	(integer) how much more it is to you to stay in, right now.
mychips1	(integer) how many chips you have left at the moment.
pot1	(integer) how much is in the pot at the moment.
roundbets	(numattable1 x 4 matrix) matrix of all past bets during this hand. roundbets[i,j] = total amount the player in seat i put in, in betting round j.
blinds1	(integer) big blind amount.
chips1	(vector of length numattable1) list of how many chips everyone has. chips1[i] = how many chips the player in seat i has left.
ind1	(integer) which seat you're in. (So, mychips1 = chips1[ind1]).
dealer1	(integer) which seat the dealer is in.
tablesleft	(integer) how many tables are left in the tournament (including yours).

Details

When this function is called from tourn1(), cards1[2,1] is always less than or equal to cards1[1,1]. If the player in the big blind seat does not have enough chips to pay the big blind, then blinds1 is still the amount that the big blind would have been. If only 2 players are left, then tourn1() uses the convention that the "dealer" is the big blind in determining dealer1.

Value

integer indicating the number of chips you are betting. 0 means fold.

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, crds1, board1,  round1, currentbet, mychips1,
```

```
  pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft){
  ## if pair of 9s or better, then all in
  a1 = 0
  if((crds1[1,1] == crds1[2,1]) && (crds1[2,1] > 8.5)) a1 = mychips1
  a1
} ## end of ursula
```

---

vera *A sample function that goes all in or folds*

Description

Goes all in with any pair, any suited cards, or if the smaller card is at least 9.

Usage

```
vera(numattable1, crds1, board1, round1, currentbet, mychips1,
pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft)
```

Arguments

numattable1	(integer) number of players at the table
crds1	(2x2 matrix) your hole cards. cards1[1,1] = the number of card 1 (between 2 and 14). cards[1,2] = suit of card 1 (between 1 and 4). cards1[2,1] = the number (2-14) of card 2. cards[2,2] = suit of card 2 (1-4).
board1	(5x2 matrix) the board cards. board1[1,1] = number of first card (2-14). board1[1,2] = suit (1-4) of first card. Both are zero if the card hasn't been seen yet. For instance, if bettinground1 < 3, then board[4,1] = board1[4,2] = 0.
round1	(integer) which betting round it is. 1 = preflop, 2 = after flop, 3 = after turn, 4 = after river.
currentbet	(integer) how much more it is to you to stay in, right now.
mychips1	(integer) how many chips you have left at the moment.
pot1	(integer) how much is in the pot at the moment.
roundbets	(numattable1 x 4 matrix) matrix of all past bets during this hand. roundbets[i,j] = total amount the player in seat i put in, in betting round j.
blinds1	(integer) big blind amount.
chips1	(vector of length numattable1) list of how many chips everyone has. chips1[i] = how many chips the player in seat i has left.
ind1	(integer) which seat you're in. (So, mychips1 = chips1[ind1]).
dealer1	(integer) which seat the dealer is in.
tablesleft	(integer) how many tables are left in the tournament (including yours).

Details

When this function is called from tourn1(), cards1[2,1] is always less than or equal to cards1[1,1]. If the player in the big blind seat does not have enough chips to pay the big blind, then blinds1 is still the amount that the big blind would have been. If only 2 players are left, then tourn1() uses the convention that the "dealer" is the big blind in determining dealer1.

Value

integer indicating the number of chips you are betting. 0 means fold.

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, crds1, board1,  round1, currentbet,  mychips1, pot1,
    roundbets, blinds1, chips1, ind1, dealer1, tablesleft){
    ## if any pair, suited anything, or if the smaller card is at least 9,
    ## then all in
    a1 = 0
   if((crds1[1,1] == crds1[2,1]) || (crds1[1,2] == crds1[2,2]) || (crds1[2,1] > 8.5)) a1 = mychips1
    a1
  } ## end of vera
```

`william` *A sample function that goes all in or folds*

Description

Goes all in if you only have less than 3 times the big blind, or have AA, or if nobody's gone all in yet, or if you have KK or QQ and less than 10 players are at the table. If the hole cards are 10 9, then it goes all in with 40% prob.

Usage

```
william(numattable1, crds1, board1, round1, currentbet, mychips1,
pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft)
```

Arguments

numattable1	(integer) number of players at the table
crds1	(2x2 matrix) your hole cards. cards1[1,1] = the number of card 1 (between 2 and 14). cards[1,2] = suit of card 1 (between 1 and 4). cards1[2,1] = the number (2-14) of card 2. cards[2,2] = suit of card 2 (1-4).
board1	(5x2 matrix) the board cards. board1[1,1] = number of first card (2-14). board1[1,2] = suit (1-4) of first card. Both are zero if the card hasn't been seen yet. For instance, if bettinground1 < 3, then board[4,1] = board1[4,2] = 0.
round1	(integer) which betting round it is. 1 = preflop, 2 = after flop, 3 = after turn, 4 = after river.
currentbet	(integer) how much more it is to you to stay in, right now.
mychips1	(integer) how many chips you have left at the moment.
pot1	(integer) how much is in the pot at the moment.
roundbets	(numattable1 x 4 matrix) matrix of all past bets during this hand. roundbets[i,j] = total amount the player in seat i put in, in betting round j.
blinds1	(integer) big blind amount.
chips1	(vector of length numattable1) list of how many chips everyone has. chips1[i] = how many chips the player in seat i has left.
ind1	(integer) which seat you're in. (So, mychips1 = chips1[ind1]).
dealer1	(integer) which seat the dealer is in.
tablesleft	(integer) how many tables are left in the tournament (including yours).

Details

When this function is called from tourn1(), cards1[2,1] is always less than or equal to cards1[1,1]. If the player in the big blind seat does not have enough chips to pay the big blind, then blinds1 is still the amount that the big blind would have been. If only 2 players are left, then tourn1() uses the convention that the "dealer" is the big blind in determining dealer1.

Value

integer indicating the number of chips you are betting. 0 means fold.

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, crds1, board1,  round1, currentbet,  mychips1, pot1,
```

```
    roundbets, blinds1, chips1, ind1, dealer1, tablesleft){
    ## if you only have less than 3 times the big blind, then all in.
    ## if AA, then all in.
    ## if T9, then all in with 40% prob.
    ## if nobody's gone all in yet, then go all in.
    ## if KK or QQ, and less than 10 players at table, then all in.
    a1 = 0
    if(mychips1 < 3*blinds1) a1 = mychips1
    if((crds1[1,1] == 14) && (crds1[2,1] == 14)) a1 = mychips1
    if((crds1[1,1] == 10) && (crds1[2,1] == 9)){
u1 = runif(1)
if(u1 < .4) a1 = mychips1
if(u1 > .4) a1 = 0
    }
    if(currentbet == blinds1) a1 = mychips1
   if((crds1[1,1] == crds1[2,1]) && (crds1[1,1] > 11.5) && (numattable1<10)) a1 = mychips1
    a1
  } ## end of william
```

---

writebets1 *Write the bets on the plot*

Description

Writes the players' bets on the current plot. Used in mygraphics1(), which in turn is used by tourn1().

Usage

```
writebets1(b9, y1, numattable1, b3, playerseats1, chips1, chips2, over1, myfast1, name1)
```

Arguments

b9

y1

numattable1

b3

playerseats1

chips1

chips2

over1

myfast1

name1

Author(s)

Frederic Paik Schoenberg

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(b9,y1,numattable1,b3,playerseats1,chips1,chips2,over1,myfast1,name1){
    # cat("\n Round ",y1,"\n",b9$il1,"\n",b9$bl1,".\n")
    cardname1 = c(as.character(1:9),"T","J","Q","K","A")
    suitname1 = c(2,3,4,6)
    drawnyet1 = rep(0,numattable1)
    if((y1 == 1) || (over1<1)){
ilen1 = length(b9$il1)
rember1 = rep(-1,numattable1)
if(ilen1 > 1.5) for(j in c(2:ilen1)){
    i = b9$il1[j]
    if((y1 == 1) && (drawnyet1[i] < 1) && (j>3.5)){
    text(10*i,60,cardname1[b3$plnum1[i,1]],col=suitname1[b3$plsuit1[i,1]],cex=2)
    text(10*i+2,60,cardname1[b3$plnum1[i,2]],col=suitname1[b3$plsuit1[i,2]],cex=2)
    drawnyet1[i] = 2
    if(myfast1<1) locator(1)
    }
    if(rember1[i] != b9$bl1[j]){
text(10*i,50-5*y1,rember1[i],col="white")
text(10*i,50-5*y1,rember1[i],col="white")
if((y1>1) || (b9$bl1[j] > .5)) text(10*i,50-5*y1,b9$bl1[j])
    }
    rember1[i] = b9$bl1[j]
    if(b9$out1[j] > 1.5){
text(10*i,60,cardname1[b3$plnum1[i,1]],col="white",cex=2)
text(10*i+2,60,cardname1[b3$plnum1[i,2]],col="white",cex=2)
text(10*i,80,as.character(name1[playerseats1[i]]),cex=1+.1*b9$i1[i],col="white")
text(10*i,50,"BETS:",col="white")
text(10*i,50,"BETS:",col="white")
if(chips1[i] == chips2[i]) text(10*i,75,paste("(",chips1[i],")"),col="white")
    }
    if((y1 > 1.5) || (j > 3.5))  if(myfast1<1) locator(1)
}
    }
  } ## end of writebets1
```

---

xena *A sample function that goes all in or folds*

Description

Goes all in with a pair of 10s or higher, or with a pair of 7s or higher if there are 6 or fewer players, or with any cards if you have less than 2 big blinds. With AK or AQ, all in with probability 75%. Or if nobody's raised yet, under certain conditions (see details).

Usage

```
xena(numattable1, crds1, board1, round1, currentbet, mychips1,
pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft)
```

Arguments

numattable1	(integer) number of players at the table
crds1	(2x2 matrix) your hole cards. cards1[1,1] = the number of card 1 (between 2 and 14). cards[1,2] = suit of card 1 (between 1 and 4). cards1[2,1] = the number (2-14) of card 2. cards[2,2] = suit of card 2 (1-4).
board1	(5x2 matrix) the board cards. board1[1,1] = number of first card (2-14). board1[1,2] = suit (1-4) of first card. Both are zero if the card hasn't been seen yet. For instance, if bettinground1 < 3, then board[4,1] = board1[4,2] = 0.
round1	(integer) which betting round it is. 1 = preflop, 2 = after flop, 3 = after turn, 4 = after river.
currentbet	(integer) how much more it is to you to stay in, right now.
mychips1	(integer) how many chips you have left at the moment.
pot1	(integer) how much is in the pot at the moment.
roundbets	(numattable1 x 4 matrix) matrix of all past bets during this hand. roundbets[i,j] = total amount the player in seat i put in, in betting round j.
blinds1	(integer) big blind amount.
chips1	(vector of length numattable1) list of how many chips everyone has. chips1[i] = how many chips the player in seat i has left.
ind1	(integer) which seat you're in. (So, mychips1 = chips1[ind1]).
dealer1	(integer) which seat the dealer is in.
tablesleft	(integer) how many tables are left in the tournament (including yours).

Details

Goes all in with a pair of 10s or higher, or with a pair of 7s or higher if there are 6 or fewer players, or with any cards if you have less than 2 big blinds. With AK or AQ, all in with probability 75%. If nobody's raised yet, ... and if there are 3 or fewer players left behind you, then go all in with any pair or any ace. ... and there's only 1 or 2 players behind you, then go all in with any cards.

When this function is called from tourn1(), cards1[2,1] is always less than or equal to cards1[1,1]. If the player in the big blind seat does not have enough chips to pay the big blind, then blinds1 is still the amount that the big blind would have been. If only 2 players are left, then tourn1() uses the convention that the "dealer" is the big blind in determining dealer1.

Value

integer indicating the number of chips you are betting. 0 means fold.

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, crds1, board1,  round1, currentbet, mychips1, pot1,
    roundbets, blinds1, chips1, ind1, dealer1, tablesleft){
    ## if pair of 10s or higher, all in for sure, no matter what.
    ## if AK or AQ, all in with probability 75%.
    ## if pair of 7s or higher and there are 6 or fewer players
    ##      at your table (including you), then all in.
    ## if your chip count is less than twice the big blind, go all in with any cards.
    ## if nobody's raised yet:
    ##     ... and if there are 3 or fewer players left behind you,
    ##              then go all in with any pair or any ace.
    ##     ... and there's only 1 or 2 players behind you,
    ##              then go all in with any cards.
    a1 = 0
    x = runif(1)               ## x is a random number between 0 and 1.
    y = max(roundbets[,1])     ## y is the maximum bet so far.
    big1 = dealer1 + 2
    if(big1 > numattable1) big1 = big1 - numattable1
    z = big1 - ind1
    if(z<0) z = z + numattable1
    ## the previous 4 lines make it so z is the number of players left to act behind you.
    if((crds1[1,1] == crds1[2,1]) && (crds1[2,1] > 9.5)) a1 = mychips1
    if((crds1[1,1] == 14) && (crds1[1,2]>11.5) && (x<.75)) a1 = mychips1
   if((crds1[1,1] == crds1[2,1]) && (crds1[2,1] > 6.5) && (numattable1 < 6.5)) a1 = mychips1
    if(mychips1 < 2*blinds1) a1 = mychips1
    if(y <= blinds1){
if((z < 3.5) && ((crds1[1,1] == crds1[2,1]) || (crds1[1,1] == 14))) a1 = mychips1
if(z < 2.5) a1 = mychips1
    }
    a1
  } ## end of xena
```

`yosef` *A sample function that does not necessarily goes all in or fold.*

Description

Goes all in with a pair of at least 9. If nobody's bet (or raised the blinds) yet, then bet (or raise the blinds) the minimum amount possible.

Usage

```
yosef(numattable1, crds1, board1, round1, currentbet, mychips1,
pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft)
```

Arguments

numattable1	(integer) number of players at the table
crds1	(2x2 matrix) your hole cards. cards1[1,1] = the number of card 1 (between 2 and 14). cards[1,2] = suit of card 1 (between 1 and 4). cards1[2,1] = the number (2-14) of card 2. cards[2,2] = suit of card 2 (1-4).
board1	(5x2 matrix) the board cards. board1[1,1] = number of first card (2-14). board1[1,2] = suit (1-4) of first card. Both are zero if the card hasn't been seen yet. For instance, if bettinground1 < 3, then board[4,1] = board1[4,2] = 0.
round1	(integer) which betting round it is. 1 = preflop, 2 = after flop, 3 = after turn, 4 = after river.
currentbet	(integer) how much more it is to you to stay in, right now.
mychips1	(integer) how many chips you have left at the moment.
pot1	(integer) how much is in the pot at the moment.
roundbets	(numattable1 x 4 matrix) matrix of all past bets during this hand. roundbets[i,j] = total amount the player in seat i put in, in betting round j.
blinds1	(integer) big blind amount.
chips1	(vector of length numattable1) list of how many chips everyone has. chips1[i] = how many chips the player in seat i has left.
ind1	(integer) which seat you're in. (So, mychips1 = chips1[ind1]).
dealer1	(integer) which seat the dealer is in.
tablesleft	(integer) how many tables are left in the tournament (including yours).

Details

When this function is called from tourn1(), cards1[2,1] is always less than or equal to cards1[1,1]. If the player in the big blind seat does not have enough chips to pay the big blind, then blinds1 is still the amount that the big blind would have been. If only 2 players are left, then tourn1() uses the convention that the "dealer" is the big blind in determining dealer1.

Value

integer indicating the number of chips you are betting. 0 means fold.

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, crds1, board1,  round1, currentbet,  mychips1, pot1,
```

```
    roundbets, blinds1, chips1, ind1, dealer1, tablesleft){
    ## if pair of at least 9, then all in
    ## if nobody's bet (or raised the blinds) yet,
    ## then bet (or raise the blinds) the minimum amount possible.
    ## note: if you're big blind, or if you've already called, then
    ## those cases have to be handled separately, since for instance if
    ## someone's raised you exactly one more big blind,
    ## then you should fold, but it will look like
    ## nobody's raised yet since it'll again be one big blind to you.
    a1 = 0
    bigb = dealer1 + 2
    if(bigb > numattable1) bigb = bigb - numattable1
    if(round1 == 1){
if((roundbets[ind1,1] < blinds1 - .5) && (currentbet < blinds1+.5)){
    a1 = min(2*blinds1, mychips1)
} else if ((ind1 == bigb) && (currentbet < blinds1 - .5)) a1 = min(blinds1, mychips1)
    }
    if((round1 > 1.5) && (currentbet < .5)) a1 = min(blinds1, mychips1)
    if((crds1[1,1] == crds1[2,1]) && (crds1[2,1] > 8.5)) a1 = mychips1
    a1
  } ## end of yosef
```

---

zelda *A sample function that does not necessarily go all in or fold.*

Description

This function bets, raises, calls, or folds depending mainly on the strength of its hand and the number of players left.

Usage

```
zelda(numattable1, crds1, board1, round1, currentbet, mychips1,
pot1, roundbets, blinds1, chips1, ind1, dealer1, tablesleft)
```

Arguments

numattable1	(integer) number of players at the table
crds1	(2x2 matrix) your hole cards. cards1[1,1] = the number of card 1 (between 2 and 14). cards[1,2] = suit of card 1 (between 1 and 4). cards1[2,1] = the number (2-14) of card 2. cards[2,2] = suit of card 2 (1-4).
board1	(5x2 matrix) the board cards. board1[1,1] = number of first card (2-14). board1[1,2] = suit (1-4) of first card. Both are zero if the card hasn't been seen yet. For instance, if bettinground1 < 3, then board[4,1] = board1[4,2] = 0.
round1	(integer) which betting round it is. 1 = preflop, 2 = after flop, 3 = after turn, 4 = after river.
currentbet	(integer) how much more it is to you to stay in, right now.

mychips1	(integer) how many chips you have left at the moment.
pot1	(integer) how much is in the pot at the moment.
roundbets	(numattable1 x 4 matrix) matrix of all past bets during this hand. roundbets[i,j] = total amount the player in seat i put in, in betting round j.
blinds1	(integer) big blind amount.
chips1	(vector of length numattable1) list of how many chips everyone has. chips1[i] = how many chips the player in seat i has left.
ind1	(integer) which seat you're in. (So, mychips1 = chips1[ind1]).
dealer1	(integer) which seat the dealer is in.
tablesleft	(integer) how many tables are left in the tournament (including yours).

Details

pre-flop: AK: Make a big raise if nobody has yet. Otherwise call. AQ: call a small raise, or make one if nobody has yet. AJ, AT, KQ, KJ, QJ: call a tiny raise. A9, KT, K9, QT, JT, T9: call a tiny raise if in late position (within 2 of the dealer). Suited A2-AJ: call a small raise. 22-99: call a small raise. TT-KK: make a huge raise. If someone's raised huge already, then go all in. AA: make a small raise. If there's been a raise already, then double how much it is to you. post-flop: If there's a pair on the board and you don't have a set, then check/call up to small bet. Same thing if there's 3-of-a-kind on the board and you don't have a full house or more. If you have top pair or an overpair or two pairs or a set, make a big bet (call any bigger bet). Otherwise, if nobody's made even a small bet yet, then with prob. 20% make a big bluff bet. If you're the last to decide and nobody's bet yet, then increase this prob. to 50%. If you have an inside straight draw or flush draw then make a small bet (call any bigger bet). If you have a straight or better, then just call. Otherwise fold. after turn: If there's a pair on the board and you don't have a set, then check/call up to small bet. Same thing if there's 3-of-a-kind on the board and you don't have a full house or more. Otherwise, if you have top pair or better, go all in. If you had top pair or overpair but now don't, then check/call a medium bet but fold to more. If you have an inside straight draw or flush draw then check/call a medium bet as well. Otherwise check/fold. after river: If there's a pair on the board and you don't have a set, then check/call up to small bet. Same thing if there's 3-of-a-kind on the board and you don't have a full house or more. Otherwise, if you have two pairs or better, go all in. If you have one pair, then check/call a small bet. With nothing, go all-in with probability 10%; otherwise check/fold.

When this function is called from tourn1(), cards1[2,1] is always less than or equal to cards1[1,1]. If the player in the big blind seat does not have enough chips to pay the big blind, then blinds1 is still the amount that the big blind would have been. If only 2 players are left, then tourn1() uses the convention that the "dealer" is the big blind in determining dealer1.

Value

integer indicating the number of chips you are betting. 0 means fold.

References

Schoenberg, F. (2011). An Introduction to Probability with Texas Holdem Examples. Taylor and Francis, New York.

Examples

```
##---- Should be DIRECTLY executable !! ----
##-- ==>  Define data, use random,
##--or do  help(data=index)  for the standard data sets.

## The function is currently defined as
function(numattable1, crds1, board1,  round1, currentbet,  mychips1, pot1,
    roundbets, blinds1, chips1, ind1, dealer1, tablesleft){
    a1 = 0 ## how much I'm gonna end up betting. Note that the default is zero.
    a2 = min(mychips1, currentbet) ## how much it costs to call

    if(round1 == 1){ ## pre-flop:
## AK: Make a big raise if nobody has yet. Otherwise call.
## AQ: call a small raise, or make one if nobody has yet.
## AJ, AT, KQ, KJ, QJ: call a tiny raise.
## A9, KT, K9, QT, JT, T9: call a tiny raise if in late position
## (within 2 of the dealer).
## Suited A2-AJ: call a small raise.
## 22-99: call a small raise.
## TT-KK: make a huge raise. If someone's raised huge already, then go all in.
## AA: make a small raise. If there's been a raise already,
## then double how much it is to you.

a3 = 2*blinds1+1 ## how much a tiny raise would be
a4 = 4*blinds1+1 ## how much a small raise would be
a5 = max(8*blinds1,mychips1/4)+1 ## how much a big raise would be
a6 = max(12*blinds1,mychips1/2)+1 ## how much a huge raise would be
a7 = dealer1 - ind1
if(a7 < -.5) a7 = a7 + numattable1 ## your position: a7 = how many hands til you're dealer

if((crds1[1,1] == 14) && (crds1[2,1] == 13)){
    a1 = max(a2,a5)
}
if((crds1[1,1] == 14) && (crds1[2,1] == 12)){
    if(a2 < a4){
a1 = a4
    } else if(a2 > a5){
a1 = 0
    } else a1 = a2
}
if(((crds1[1,1] == 14) && ((crds1[2,1] < 11.5) && (crds1[2,1] > 9.5))) ||
    ((crds1[1,1] == 13) && (crds1[2,1] > 10.5)) ||
    ((crds1[1,1] == 12) && (crds1[2,1] == 11))){
    if(a2 < a3) a1 = a2
}
if(((crds1[1,1] == 14) && (crds1[2,1] == 9)) ||
    ((crds1[1,1] == 13) && ((crds1[2,1] == 10) || (crds1[2,1] == 9))) ||
    ((crds1[1,1] == 12) && (crds1[2,1] == 10)) ||
    ((crds1[1,1] == 11) && (crds1[2,1] == 10)) ||
    ((crds1[1,1] == 10) && (crds1[2,2] == 9))){
    if((a2 < a3) && (a7<2.5)) a1 = a2
}
```

```
if((crds1[1,2] == crds1[2,2]) && (crds1[1,1] == 14) && (crds1[2,1] < 11.5)){
    if(a2<a4) a1 = a2
    ## Note: this trumps the previous section, since it comes later in the code.
}
if((crds1[1,1] == crds1[2,1])){ ## pairs:
    if(crds1[1,1] < 9.5){
if(a2 < a4) a1 = a2
    } else if(crds1[1,1] < 13.5){
if(a2<a5) a1 = a5 else a1 = mychips1
    } else {
if(a2 < blinds1 + .5) a1 = a4 else a1 = min(2*a2,mychips1)
    }
}
    }
    if(round1 == 2){ ## post-flop:
## If there's a pair on the board and you don't have a set, then check/call up to small bet.
## Same thing if there's 3-of-a-kind on the board and you don't have a full house or more.
## If you have top pair or an overpair or two pairs or a set,
## make a big bet (call any bigger bet).
## Otherwise, if nobody's made even a small bet yet,
## then with prob. 20% make a big bluff bet.
## If you're the last to decide and nobody's bet yet, then increase this prob. to 50%.
## If you have an inside straight draw or flush draw then make
## a small bet (call any bigger bet).
## If you have a straight or better, then just call.
## Otherwise fold.

a5 = min(sum(roundbets[,1]),mychips1) ## how much
## a big bet would be (prev round's pot size)
a6 = min(.5*sum(roundbets[,1]),mychips1) ## how much a small bet would be
x = handeval(c(crds1[1:2,1], board1[1:3,1]),
c(crds1[1:2,2], board1[1:3,2])) ## what you have
x1 = handeval(c(board1[1:3,1]),c(board1[1:3,2])) ## what's on the board
y = straightdraw1(c(crds1[1:2,1], board1[1:3,1]))
z = flushdraw1(c(crds1[1:2,2], board1[1:3,2]))
topcard1 = max(board1[1:3,1])
a7 = runif(1) ## random number uniformly distributed between 0 and 1
a8 = (1:numattable1)[roundbets[,1] == roundbets[ind1,1]] ## others who can still bet
## The next 5 lines may seem weird, but the purpose is explained in the next comment:
a9 = a8 - dealer1
for(i in 1:length(a9)) if(a9[i]<.5) a9[i] = a9[i] + numattable1
a10 = ind1 - dealer1
if(a10 < .5) a10 = a10 + numattable1
a11 = 2*(a10 == max(a9))   ## So a11 = 2 if you're last to decide; otherwise a11 = 0.

if((x1 > 1000000) && (x < 3000000)){
    if(a2 < a6) a1 = a2
} else if((x1 > 3000000) && (x < 6000000)){
    if(a2 < a6) a1 = a2
} else if(x > 1000000 + 15^3*topcard1){
    a1 = max(a5,a2)
} else if((a2 < a6) && ((a7 < .20) || ((a7 < .50) && (a11>1)))){
    a1 = a6
```

```
}
if((y == 4) || (z == 4)) a1 = max(a6, a2)
if(x > 4000000) a1 = a2
    }
    if(round1 == 3){ ## after turn:
## If there's a pair on the board and you don't have a set, then check/call up to small bet.
## Same thing if there's 3-of-a-kind on the board and you don't have a full house or more.
## Otherwise, if you have top pair or better, go all in.
## If you had top pair or overpair but now don't, then check/call a medium bet
## but fold to more.
## If you have an inside straight draw or flush draw then check/call a medium bet as well.
## Otherwise check/fold.
a6 = min(1/3*sum(roundbets[,1:2]),mychips1) ## small bet (1/3 of prev round's pot size)
a5 = min(.75*sum(roundbets[,1:2]),mychips1) ## medium bet (3/4 of prev round's pot size)
x = handeval(c(crds1[1:2,1], board1[1:4,1]),
c(crds1[1:2,2], board1[1:4,2])) ## what you have
x1 = handeval(c(board1[1:4,1]),c(board1[1:4,2])) ## what's on the board
y = straightdraw1(c(crds1[1:2,1], board1[1:4,1]))
z = flushdraw1(c(crds1[1:2,2], board1[1:4,2]))
topcard1 = max(board1[1:4,1])
oldtopcard1 = max(board1[1:3,1])
if((x1 > 1000000) && (x < 3000000)){
    if(a2 < a6) a1 = a2
} else if((x1 > 3000000) && (x < 6000000)){
    if(a2 < a6) a1 = a2
} else if(x > 1000000 + 15^3*topcard1){
    a1 = mychips1
} else if(x > 1000000 + 15^3*oldtopcard1){
    if(a2 < a5) a1 = a2
} else if((y == 4) || (z == 4)){
    if(a2 < a5) a1 = a2
}
    }
    if(round1 == 4){ ## after river:
## If there's a pair on the board and you don't have a set, then check/call up to small bet.
## Same thing if there's 3-of-a-kind on the board and you don't have a full house or more.
## Otherwise, if you have two pairs or better, go all in.
## If you have one pair, then check/call a small bet.
## With nothing, go all-in with probability 10%; otherwise check/fold.
a6 = .45+runif(1)/10  ## random number between .45 and .55
a5 = min(a6*sum(roundbets[,1:3]),mychips1) ## small bet:
## around 1/2 of pot size; VARIES RANDOMLY
x = handeval(c(crds1[1:2,1], board1[1:5,1]), c(crds1[1:2,2], board1[1:5,2]))
x1 = handeval(c(board1[1:5,1]),c(board1[1:5,2])) ## what's on the board
if((x1 > 1000000) && (x < 3000000)){
    if(a2 < a5) a1 = a2
} else if((x1 > 3000000) && (x < 6000000)){
    if(a2 < a5) a1 = a2
} else if(x > 2000000){
    a1 = mychips1
} else if(x > 1000000){
    if(a2 < a5) a1 = a2
} else if(runif(1)<.10){
```

```
    a1 = mychips1
}
    }
    round(a1)
  } ## end of zelda
```

Index