If you are passionate about cricket, and love analyzing cricket performances, then check out my 2 racy books on cricket! In my books, I perform detailed yet compact analysis of performances of both batsmen, bowlers besides evaluating team & match performances in Tests , ODIs, T20s & IPL. You can buy my books on cricket from Amazon at $12.99 for the paperback and $4.99/$6.99 respectively for the kindle versions. The books can be accessed at [Cricket analytics with cricketr](https://www.amazon.com/s/ref=nb_sb_noss?url=search-alias%3Daps&field-keywords=Cricket+analytics+with+cricketr%3A+Third+edition&rh=i%3Aaps%2Ck%3ACricket+analytics+with+cricketr%3A+Third+edition)  and [Beaten by sheer pace-Cricket analytics with yorkr](https://www.amazon.com/dp/1973561719) A must read for any cricket lover! Check it out!!



While ML & DL are very useful and interesting to make inferences and predictions of outputs from input variables, optimization computes the choice of input which results in maximizing or minimizing the output. So I made a small course correction and started on a course from India’s own NPTEL [Introduction to Linear Programming](http://freevideolectures.com/Course/2365/Fundamentals-of-Operations-Research) by Prof G. Srinivasan of IIT Madras (highly recommended!). The lectures are delivered with remarkable clarity by the Prof and I am just about halfway through the course (each lecture is of 50-55 min duration), when I decided that I needed to try to formulate and solve some real world Linear Programming problem.

As usual, I turned towards cricket for some appropriate situations, and sure enough it was there in the open. For this LP formulation I take International T20 and IPL, though International ODI will also work equally well.  You can download the associated code and data for this from Github at [LP-cricket-analysis](https://github.com/tvganesh/LP-cricket-analysis)

In T20 matches the captain has to make choice of how to rotate bowlers with the aim of restricting the batting side. Conversely, the batsmen need to take advantage of the bowling strength to maximize the runs scored.

**Note**:  
a) A simple and obvious strategy would be  
– If the ith bowler’s economy rate is less than the economy rate of the jth bowler i.e.  
er_{i}< er_{j}then have bowler ‘i’ to bowl more overs as his/her economy rate is better

b)A better strategy would be to consider the economy rate of each bowler against each batsman. How often  have we witnessed bowlers with a great bowling average get thrashed time and again by the same batsman, or a bowler who is generally very poor being very effective against a particular batsman. i.e. er_{ij}< er_{ik}where the jth bowler is more effective than the kth bowler against the ith batsman. This now becomes a linear optimization problem as we can have several combinations of *number of overs* X *economy rate* for different bowlers and we will have to solve this algorithmically to determine the lowest score for bowling performance or highest score for batting order.

This post uses the latter approach to optimize bowling change and batting lineup.

Let is take a hypothetical situation  
Assume there are 3 bowlers – bwlr_{1},bwlr_{2},bwlr_{3}  
and there are 4 batsmen – bman_{1},bman_{2},bman_{3},bman_{4}

Let the economy rate er_{ij}be the Economy Rate of the jth bowler to the ith batsman. Also if remaining overs for the bowlers are o_{1},o_{2},o_{3}  
and the total number of overs left to be bowled are  
o_{1}+o_{2}+o_{3} = Nthen the question is

a) Given the economy rate of each bowler per batsman, how many overs should each bowler bowl, so that the total runs scored by all the batsmen are minimum?

b) Alternatively, if the know the individual strike rate of a batsman against the individual bowlers, how many overs should each batsman face with a bowler so that the total runs scored is maximized?

**1. LP Formulation for bowling order**

Let the economy rate er_{ij}be the Economy Rate of the jth bowler to the ith batsman.  
**Objective function** : Minimize –  
  
i.e.  
\sum_{i=1}^{i=m}\sum_{j=1}^{i=n}er_{ij}*o_{ij}  
**Constraints**  
Where o_{j}is the number of overs remaining for the jth bowler against  ‘k’ batsmen  
o_{j1} + o_{j2} + .. o_{jk} < o_{j}  
and if the total number of overs remaining to be bowled is N then  
o_{1} + o_{2} +...+ o_{k} = Nor  
\sum_{j=1}^{j=k} o_{j} =N  
The overs that any bowler can bowl is o_{j} >=0

**2. LP Formulation for batting lineup**

Let the strike rate sr_{ij} be the Strike Rate of the ith batsman to the jth bowler  
**Objective function** : Maximize –  
  
i.e.  
\sum_{i=1}^{i=4}\sum_{j=1}^{i=3}sr_{ij}*o_{ij}  
**Constraints**  
Where o_{j}is the number of overs remaining for the jth bowler against  ‘k’ batsmen  
o_{j1} + o_{j2} + .. o_{jk} < o_{j}  
and the total number of overs remaining to be bowled is N then  
o_{1} + o_{2} +...+ o_{k} = Nor  
\sum_{j=1}^{j=k} o_{j} =N  
The overs that any bowler can bowl is  
o_{j} >=0

**lpSolveAPI**– For this maximization and minimization problem I used lpSolveAPI.

Below I take 2 simple examples (example1 & 2)  to ensure that my LP formulation and solution is correct before applying it on real T20 cricket data (Intl. T20 and IPL)

**3. LP formulation (Example 1)**

Initially I created a test example to ensure that I get the LP formulation and solution correct. Here the er1=4 and er2=3 and o1 & o2 are the overs bowled by bowlers 1 & 2. Also o1+o2=4 In this example as below

o1 o2 Obj Fun(=4o*1+3o*2)  
1    3      13  
2    2      14  
3    1      15

library(lpSolveAPI)

library(dplyr)

library(knitr)

lprec <- make.lp(0, 2)

a <-lp.control(lprec, sense="min")

set.objfn(lprec, c(4, 3)) # Economy Rate of 4 and 3 for er1 and er2

add.constraint(lprec, c(1, 1), "=",4) # o1 + o2 =4

add.constraint(lprec, c(1, 0), ">",1) # o1 > 1

add.constraint(lprec, c(0, 1), ">",1) # o2 > 1

lprec

## Model name:

## C1 C2

## Minimize 4 3

## R1 1 1 = 4

## R2 1 0 >= 1

## R3 0 1 >= 1

## Kind Std Std

## Type Real Real

## Upper Inf Inf

## Lower 0 0

b <-solve(lprec)

get.objective(lprec) # 13

## [1] 13

get.variables(lprec) # 1 3

## [1] 1 3

**Note 1:** In the above example **13 runs** is the minimum that can be scored and this requires

**LP solution**:  
Minimum runs=13

* o1=1
* o2=3

**Note 2:**The numbers in the columns represent the number of overs that need to be bowled by a bowler to the corresponding batsman.

**4. LP formulation (Example 2)**

In this formulation there are 2 bowlers and 2 batsmen o11,o12 are the oves bowled by bowler 1 to batsmen 1 & 2 and o21, o22 are the overs bowled by bowler 2 to batsmen 1 & 2 er11=4, er12=2,er21=2,er22=5 o11+o12+o21+o22=5

The solution for this manually computed is o11, o12, o21, o22 Runs  
where B11, B12 are the overs bowler 1 bowls to batsman 1 and B21 and B22 are overs bowler 2 bowls to batsman 2

o11     o12    o21    o22      Runs=(4\*o11+2\*o12+2\*o21+5\*o22)  
1            1             1            2           18  
1           2              1             1           15  
2           1              1            1            17  
1           1               2            1            15

lprec <- make.lp(0, 4)

a <-lp.control(lprec, sense="min")

set.objfn(lprec, c(4, 2,2,5))

add.constraint(lprec, c(1, 1,0,0), "<=",8)

add.constraint(lprec, c(0, 0,1,1), "<=",7)

add.constraint(lprec, c(1, 1,1,1), "=",5)

add.constraint(lprec, c(1, 0,0,0), ">",1)

add.constraint(lprec, c(0, 1,0,0), ">",1)

add.constraint(lprec, c(0, 0,1,0), ">",1)

add.constraint(lprec, c(0, 0,0,1), ">",1)

lprec

## Model name:

## C1 C2 C3 C4

## Minimize 4 2 2 5

## R1 1 1 0 0 <= 8

## R2 0 0 1 1 <= 7

## R3 1 1 1 1 = 5

## R4 1 0 0 0 >= 1

## R5 0 1 0 0 >= 1

## R6 0 0 1 0 >= 1

## R7 0 0 0 1 >= 1

## Kind Std Std Std Std

## Type Real Real Real Real

## Upper Inf Inf Inf Inf

## Lower 0 0 0 0

b<-solve(lprec)

get.objective(lprec)

## [1] 15

get.variables(lprec)

## [1] 1 2 1 1

Note: In the above example **15 runs** is the minimum that can be scored and this requires

**LP Solution**:  
Minimum runs=15

* o11=1
* o12=2
* o21=1
* o22=1

It is possible to keep the minimum to other values and solves also.

**5. LP formulation for International T20 India vs Australia (Batting lineup)**

To analyze batting and bowling lineups in the cricket world I needed to get the ball-by-ball details of runs scored by each batsman against each of the bowlers. Fortunately I had already created this with my R package [yorkr](https://cran.r-project.org/web/packages/yorkr/index.html). yorkr processes yaml data from [Cricsheet](https://cricsheet.org/). So I copied the data of all matches between Australia and India in International T20s. You can download my processed data for International T20 at [Inswinger](https://github.com/tvganesh/inswinger)

load("Australia-India-allMatches.RData")

dim(matches)

## [1] 3541 25

The following functions compute the ‘Strike Rate’ of a batsman as

SR=1/overs∗∑RunsScored

Also the Economy Rate is computed as

ER=1/overs∗∑RunsConceded

Incidentally the SR=ER

# Compute the Strike Rate of the batsman

computeSR <- function(batsman1,bowler1){

a <- matches %>% filter(batsman==batsman1 & bowler==bowler1)

a1 <- a %>% summarize(totalRuns=sum(runs),count=n()) %>% mutate(SR=(totalRuns/count)\*6)

a1

}

# Compute the Economy Rate of the batsman

computeER <- function(batsman1,bowler1){

a <- matches %>% filter(batsman==batsman1 & bowler==bowler1)

a1 <- a %>% summarize(totalRuns=sum(runs),count=n()) %>% mutate(ER=(totalRuns/count)\*6)

a1

}

Here I compute the Strike Rate of Virat Kohli, Yuvraj Singh and MS Dhoni against Shane Watson, Brett Lee and MA Starc

# Kohli

kohliWatson<- computeSR("V Kohli","SR Watson")

kohliWatson

## totalRuns count SR

## 1 45 37 7.297297

kohliLee <- computeSR("V Kohli","B Lee")

kohliLee

## totalRuns count SR

## 1 10 7 8.571429

kohliStarc <- computeSR("V Kohli","MA Starc")

kohliStarc

## totalRuns count SR

## 1 11 9 7.333333

# Yuvraj

yuvrajWatson<- computeSR("Yuvraj Singh","SR Watson")

yuvrajWatson

## totalRuns count SR

## 1 24 22 6.545455

yuvrajLee <- computeSR("Yuvraj Singh","B Lee")

yuvrajLee

## totalRuns count SR

## 1 12 7 10.28571

yuvrajStarc <- computeSR("Yuvraj Singh","MA Starc")

yuvrajStarc

## totalRuns count SR

## 1 12 8 9

# MS Dhoni

dhoniWatson<- computeSR("MS Dhoni","SR Watson")

dhoniWatson

## totalRuns count SR

## 1 33 28 7.071429

dhoniLee <- computeSR("MS Dhoni","B Lee")

dhoniLee

## totalRuns count SR

## 1 26 20 7.8

dhoniStarc <- computeSR("MS Dhoni","MA Starc")

dhoniStarc

## totalRuns count SR

## 1 11 8 8.25

When we consider the batting lineup, the problem is one of maximization. In the LP formulation below V Kohli has a SR of 7.29, 8.57, 7.33 against Watson, Lee & Starc  
Yuvraj has a SR of 6.5, 10.28, 9 against Watson, Lee & Starc  
and Dhoni has a SR of 7.07, 7.8,  8.25 against Watson, Lee and Starc

The constraints are Watson, Lee and Starc have 3, 4 & 3 overs remaining respectively. The total number of overs remaining to be bowled is 9.The other constraints could be that a bowler bowls at least 1 over etc.

Formulating and solving

# 3 batsman x 3 bowlers

lprec <- make.lp(0, 9)

# Maximization

a<-lp.control(lprec, sense="max")

# Set the objective function

set.objfn(lprec, c(kohliWatson$SR, kohliLee$SR,kohliStarc$SR,

yuvrajWatson$SR,yuvrajLee$SR,yuvrajStarc$SR,

dhoniWatson$SR,dhoniLee$SR,dhoniStarc$SR))

#Assume the bowlers have 3,4,3 overs left respectively

add.constraint(lprec, c(1, 1,1,0,0,0, 0,0,0), "<=",3)

add.constraint(lprec, c(0,0,0,1,1,1,0,0,0), "<=",4)

add.constraint(lprec, c(0,0,0,0,0,0,1,1,1), "<=",3)

#o11+o12+o13+o21+o22+o23+o31+o32+o33=8 (overs remaining)

add.constraint(lprec, c(1,1,1,1,1,1,1,1,1), "=",9)

add.constraint(lprec, c(1,0,0,0,0,0,0,0,0), ">=",1) #o11 >=1

add.constraint(lprec, c(0,1,0,0,0,0,0,0,0), ">=",0) #o12 >=0

add.constraint(lprec, c(0,0,1,0,0,0,0,0,0), ">=",0) #o13 >=0

add.constraint(lprec, c(0,0,0,1,0,0,0,0,0), ">=",1) #o21 >=1

add.constraint(lprec, c(0,0,0,0,1,0,0,0,0), ">=",1) #o22 >=1

add.constraint(lprec, c(0,0,0,0,0,1,0,0,0), ">=",0) #o23 >=0

add.constraint(lprec, c(0,0,0,0,0,0,1,0,0), ">=",1) #o31 >=1

add.constraint(lprec, c(0,0,0,0,0,0,0,1,0), ">=",0) #o32 >=0

add.constraint(lprec, c(0,0,0,0,0,0,0,0,1), ">=",0) #o33 >=0

lprec

## Model name:

## a linear program with 9 decision variables and 13 constraints

b <-solve(lprec)

get.objective(lprec) #

## [1] 77.16418

get.variables(lprec) #

## [1] 1 2 0 1 3 0 1 0 1

This shows that the maximum runs that can be scored for the current strike rate is 77.16   runs in 9 overs The breakup is as follows

This is also shown below

get.variables(lprec) #

## [1] 1 2 0 1 3 0 1 0 1

This is also shown below

e <- as.data.frame(rbind(c(1,2,0,3),c(1,3,0,4),c(1,0,1,2)))

names(e) <- c("S Watson","B Lee","MA Starc","Overs")

rownames(e) <- c("Kohli","Yuvraj","Dhoni")

e

**LP Solution**:  
Maximum runs that can be scored by India against Australia is:77.164 if the 9 overs to be faced by the batsman are as below

## S Watson B Lee MA Starc Overs

## Kohli 1 2 0 3

## Yuvraj 1 3 0 4

## Dhoni 1 0 1 2

#Total overs=9

**Note:** This assumes that the batsmen perform at their current Strike Rate. Howvever anything can happen in a real game, but nevertheless this is a fairly reasonable estimate of the performance

**Note 2:**The numbers in the columns represent the number of overs that need to be bowled by a bowler to the corresponding batsman.

**Note 3:***You could try other combinations of overs for the above SR. For the above constraints 77.16 is the highest score for the given number of overs*

**6. LP formulation for International T20 India vs Australia (Bowling lineup)**

For this I compute how the bowling should be rotated between R Ashwin, RA Jadeja and JJ Bumrah when taking into account their performance against batsmen like Shane Watson, AJ Finch and David Warner. For the bowling performance I take the Economy rate of the bowlers. The data is the same as above

computeSR <- function(batsman1,bowler1){

a <- matches %>% filter(batsman==batsman1 & bowler==bowler1)

a1 <- a %>% summarize(totalRuns=sum(runs),count=n()) %>% mutate(SR=(totalRuns/count)\*6)

a1

}

# RA Jadeja

jadejaWatson<- computeER("SR Watson","RA Jadeja")

jadejaWatson

## totalRuns count ER

## 1 60 29 12.41379

jadejaFinch <- computeER("AJ Finch","RA Jadeja")

jadejaFinch

## totalRuns count ER

## 1 36 33 6.545455

jadejaWarner <- computeER("DA Warner","RA Jadeja")

jadejaWarner

## totalRuns count ER

## 1 23 11 12.54545

# Ashwin

ashwinWatson<- computeER("SR Watson","R Ashwin")

ashwinWatson

## totalRuns count ER

## 1 41 26 9.461538

ashwinFinch <- computeER("AJ Finch","R Ashwin")

ashwinFinch

## totalRuns count ER

## 1 63 36 10.5

ashwinWarner <- computeER("DA Warner","R Ashwin")

ashwinWarner

## totalRuns count ER

## 1 38 28 8.142857

# JJ Bunrah

bumrahWatson<- computeER("SR Watson","JJ Bumrah")

bumrahWatson

## totalRuns count ER

## 1 22 20 6.6

bumrahFinch <- computeER("AJ Finch","JJ Bumrah")

bumrahFinch

## totalRuns count ER

## 1 25 19 7.894737

bumrahWarner <- computeER("DA Warner","JJ Bumrah")

bumrahWarner

## totalRuns count ER

## 1 2 4 3

As can be seen from above RA Jadeja has a ER of 12.4, 6.54, 12.54 against Watson, AJ Finch and Warner also Ashwin has a ER of 9.46, 10.5, 8.14 against Watson, Finch and Warner. Similarly Bumrah has an ER of 6.6,7.89, 3 against Watson, Finch and Warner  
The constraints are Jadeja, Ashwin and Bumrah have 4, 3 & 4 overs remaining and the total overs remaining to be bowled is 10.

Formulating solving the bowling lineup is shown below

lprec <- make.lp(0, 9)

a <-lp.control(lprec, sense="min")

# Set the objective function

set.objfn(lprec, c(jadejaWatson$ER, jadejaFinch$ER,jadejaWarner$ER,

ashwinWatson$ER,ashwinFinch$ER,ashwinWarner$ER,

bumrahWatson$ER,bumrahFinch$ER,bumrahWarner$ER))

add.constraint(lprec, c(1, 1,1,0,0,0, 0,0,0), "<=",4) # Jadeja has 4 overs

add.constraint(lprec, c(0,0,0,1,1,1,0,0,0), "<=",3) # Ashwin has 3 overs left

add.constraint(lprec, c(0,0,0,0,0,0,1,1,1), "<=",4) # Bumrah has 4 overs left

add.constraint(lprec, c(1,1,1,1,1,1,1,1,1), "=",10) # Total overs = 10

add.constraint(lprec, c(1,0,0,0,0,0,0,0,0), ">=",1)

add.constraint(lprec, c(0,1,0,0,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,1,0,0,0,0,0,0), ">=",1)

add.constraint(lprec, c(0,0,0,1,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,1,0,0,0,0), ">=",1)

add.constraint(lprec, c(0,0,0,0,0,1,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,0,0,1,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,0,0,0,1,0), ">=",1)

add.constraint(lprec, c(0,0,0,0,0,0,0,0,1), ">=",0)

lprec

## Model name:

## a linear program with 9 decision variables and 13 constraints

b <-solve(lprec)

get.objective(lprec) #

## [1] 73.58775

get.variables(lprec) #

## [1] 1 2 1 0 1 1 0 1 3

The minimum runs that will be conceded by these 3 bowlers in 10 overs is 73.58 assuming the bowling is rotated as follows

e <- as.data.frame(rbind(c(1,0,0),c(2,1,1),c(1,1,3),c(4,2,4)))

names(e) <- c("RA Jadeja","R Ashwin","JJ Bumrah")

rownames(e) <- c("S Watson","AJ Finch","DA Warner","Overs")

e

**LP Solution**:  
Minimum runs that will be conceded by India against Australia is 73.58 in 10 overs if the overs bowled are as follows

## RA Jadeja R Ashwin JJ Bumrah

## S Watson 1 0 0

## AJ Finch 2 1 1

## DA Warner 1 1 3

## Overs 4 2 4

#Total overs=10

**7. LP formulation for IPL (Mumbai Indians – Kolkata Knight Riders – Bowling lineup)**

As in the case of International T20s I also have processed IPL data derived from my R package yorkr. [yorkr](https://cran.r-project.org/web/packages/yorkr/index.html). yorkr processes yaml data from [Cricsheet](https://cricsheet.org/). The processed data for all IPL matches can be downloaded from [GooglyPlus](https://github.com/tvganesh/GooglyPlus)

load("Mumbai Indians-Kolkata Knight Riders-allMatches.RData")

dim(matches)

## [1] 4237 25

# Compute the Economy Rate of the Mumbai Indian bowlers against Kolkata Knight Riders

# Gambhir

gambhirMalinga <- computeER("G Gambhir","SL Malinga")

gambhirHarbhajan <- computeER("G Gambhir","Harbhajan Singh")

gambhirPollard <- computeER("G Gambhir","KA Pollard")

#Yusuf Pathan

yusufMalinga <- computeER("YK Pathan","SL Malinga")

yusufHarbhajan <- computeER("YK Pathan","Harbhajan Singh")

yusufPollard <- computeER("YK Pathan","KA Pollard")

#JH Kallis

kallisMalinga <- computeER("JH Kallis","SL Malinga")

kallisHarbhajan <- computeER("JH Kallis","Harbhajan Singh")

kallisPollard <- computeER("JH Kallis","KA Pollard")

#RV Uthappa

uthappaMalinga <- computeER("RV Uthappa","SL Malinga")

uthappaHarbhajan <- computeER("RV Uthappa","Harbhajan Singh")

uthappaPollard <- computeER("RV Uthappa","KA Pollard")

Here

gambhirMalinga, yusufMalinga, kallisMalinga, uthappaMalinga is the ER of Malinga against Gambhir, Yusuf Pathan, Kallis and Uthappa  
gambhirHarbhajan, yusufHarbhajan, kallisHarbhajan, uthappaHarbhajan is the ER of Harbhajan against Gambhir, Yusuf Pathan, Kallis and Uthappa  
gambhirPollard, yusufPollard, kallisPollard, uthappaPollard is the ER of Kieron Pollard against Gambhir, Yusuf Pathan, Kallis and Uthappa

The constraints are Malinga, Harbhajan and Pollard have 4 overs each and remaining overs to be bowled is 10.

Formulating and solving this for the bowling lineup of Mumbai Indians against Kolkata Knight Riders

library("lpSolveAPI")

lprec <- make.lp(0, 12)

a=lp.control(lprec, sense="min")

set.objfn(lprec, c(gambhirMalinga$ER, yusufMalinga$ER,kallisMalinga$ER,uthappaMalinga$ER,

gambhirHarbhajan$ER,yusufHarbhajan$ER,kallisHarbhajan$ER,uthappaHarbhajan$ER,

gambhirPollard$ER,yusufPollard$ER,kallisPollard$ER,uthappaPollard$ER))

add.constraint(lprec, c(1,1,1,1, 0,0,0,0, 0,0,0,0), "<=",4)

add.constraint(lprec, c(0,0,0,0,1,1,1,1,0,0,0,0), "<=",4)

add.constraint(lprec, c(0,0,0,0,0,0,0,0,1,1,1,1), "<=",4)

add.constraint(lprec, c(1,1,1,1,1,1,1,1,1,1,1,1), "=",10)

add.constraint(lprec, c(1,0,0,0,0,0,0,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,1,0,0,0,0,0,0,0,0,0,0), ">=",1)

add.constraint(lprec, c(0,0,1,0,0,0,0,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,1,0,0,0,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,1,0,0,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,0,1,0,0,0,0,0,0), ">=",1)

add.constraint(lprec, c(0,0,0,0,0,0,1,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,0,0,0,1,0,0,0,0), ">=",1)

add.constraint(lprec, c(0,0,0,0,0,0,0,0,1,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,0,0,0,0,0,1,0,0), ">=",1)

add.constraint(lprec, c(0,0,0,0,0,0,0,0,0,0,1,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,0,0,0,0,0,0,0,1), ">=",0)

lprec

## Model name:

## a linear program with 12 decision variables and 16 constraints

b=solve(lprec)

get.objective(lprec) #

## [1] 55.57887

get.variables(lprec) #

## [1] 3 1 0 0 0 1 0 1 3 1 0 0

e <- as.data.frame(rbind(c(3,1,0,0,4),c(0, 1, 0,1,2),c(3, 1, 0,0,4)))

names(e) <- c("Gambhir","Yusuf","Kallis","Uthappa","Overs")

rownames(e) <- c("Malinga","Harbhajan","Pollard")

e

**LP Solution**: Mumbai Indians can restrict Kolkata Knight Riders to 55.87 in 10 overs  
if the overs are bowled as below

## Gambhir Yusuf Kallis Uthappa Overs

## Malinga 3 1 0 0 4

## Harbhajan 0 1 0 1 2

## Pollard 3 1 0 0 4

#Total overs=10

**8. LP formulation for IPL (Mumbai Indians – Kolkata Knight Riders – Batting lineup)**

As I mentioned it is possible to perform a maximation with the same formulation since computeSR<==>computeER

This just flips the problem around and computes the maximum runs that can be scored for the batsman’s Strike rate (this is same as the bowler’s Economy rate) i.e.

gambhirMalinga, yusufMalinga, kallisMalinga, uthappaMalinga is the SR of Gambhir, Yusuf Pathan, Kallis and Uthappa against Malinga  
gambhirHarbhajan, yusufHarbhajan, kallisHarbhajan, uthappaHarbhajan is the SR of Gambhir, Yusuf Pathan, Kallis and Uthappa against Harbhajan  
gambhirPollard, yusufPollard, kallisPollard, uthappaPollard is the SR of Gambhir, Yusuf Pathan, Kallis and Uthappa against Kieron Pollard.

The constraints are Malinga, Harbhajan and Pollard have 4 overs each and remaining overs to be bowled is 10.

library("lpSolveAPI")

lprec <- make.lp(0, 12)

a=lp.control(lprec, sense="max")

a <-set.objfn(lprec, c(gambhirMalinga$ER, yusufMalinga$ER,kallisMalinga$ER,uthappaMalinga$ER,

gambhirHarbhajan$ER,yusufHarbhajan$ER,kallisHarbhajan$ER,uthappaHarbhajan$ER,

gambhirPollard$ER,yusufPollard$ER,kallisPollard$ER,uthappaPollard$ER))

add.constraint(lprec, c(1,1,1,1, 0,0,0,0, 0,0,0,0), "<=",4)

add.constraint(lprec, c(0,0,0,0,1,1,1,1,0,0,0,0), "<=",4)

add.constraint(lprec, c(0,0,0,0,0,0,0,0,1,1,1,1), "<=",4)

add.constraint(lprec, c(1,1,1,1,1,1,1,1,1,1,1,1), "=",11)

add.constraint(lprec, c(1,0,0,0,0,0,0,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,1,0,0,0,0,0,0,0,0,0,0), ">=",1)

add.constraint(lprec, c(0,0,1,0,0,0,0,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,1,0,0,0,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,1,0,0,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,0,1,0,0,0,0,0,0), ">=",1)

add.constraint(lprec, c(0,0,0,0,0,0,1,0,0,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,0,0,0,1,0,0,0,0), ">=",1)

add.constraint(lprec, c(0,0,0,0,0,0,0,0,1,0,0,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,0,0,0,0,0,1,0,0), ">=",1)

add.constraint(lprec, c(0,0,0,0,0,0,0,0,0,0,1,0), ">=",0)

add.constraint(lprec, c(0,0,0,0,0,0,0,0,0,0,0,1), ">=",0)

lprec

## Model name:

## a linear program with 12 decision variables and 16 constraints

b=solve(lprec)

get.objective(lprec) #

## [1] 94.22649

get.variables(lprec) #

## [1] 0 3 0 0 0 1 0 3 0 1 3 0

e <- as.data.frame(rbind(c(0,3,0,0,3),c(0, 1, 0,3,4),c(0, 1, 3,0,4)))

names(e) <- c("Gambhir","Yusuf","Kallis","Uthappa","Overs")

rownames(e) <- c("Malinga","Harbhajan","Pollard")

e

**LP Solution**: Kolkata Knight Riders can score a maximum of 94.22 in 11 overs against Mumbai Indians  
if the the number of overs KKR face is as below

## Gambhir Yusuf Kallis Uthappa Overs

## Malinga 0 3 0 0 3

## Harbhajan 0 1 0 3 4

## Pollard 0 1 3 0 4

#Total overs=11

**Conclusion:** It is possible to thus determine the optimum no of overs to give to a specific bowler based on his/her Economy Rate with a particular batsman. Similarly one can determine the maximum runs that can be scored by a batsmen based on their strike rate with bowlers. Cricket like many other games is a game of strategy, skill, talent and some amount of luck. So while the LP formulation can provide some direction,  one must be aware anything could happen in a game of cricket!