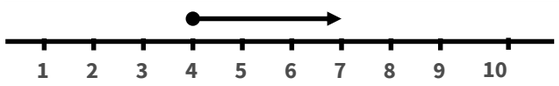
Multi-stage deterministic linkages and case definitions with diyar: : **CHEAT SHEET**

number_line objects

- A range of real numbers on a number line

```
number_line(4, 7)
```



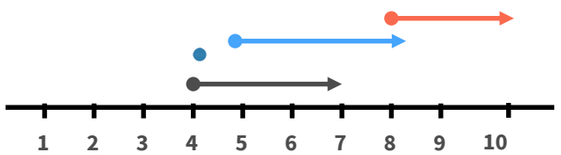
- Also supports objects that can be coerced to numeric values

```
dates <- c("30/04/2019", "10/04/2019")
dates <- as.Date(dates, "%d/%m/%Y")
number_line(dates[1], dates[2])
```

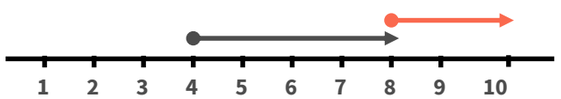
COMBINE NUMBER LINE OBJECTS

Overlapping number_line objects can be merged vertically

```
n1 <- number_line(4,7)
n2 <- as.number_line(4)
n3 <- number_line(5,8)
n4 <- number_line(8,10)
nl <- c(n1, n2, n3, n4)
```



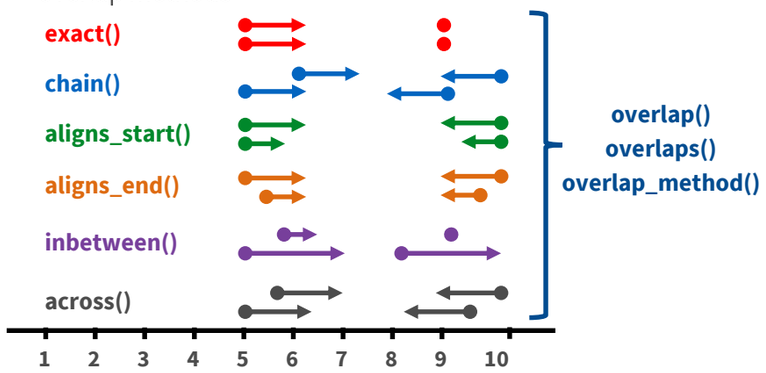

```
compress_number_line(nl)
```




```
compress_number_line(nl, collapse =T)
```

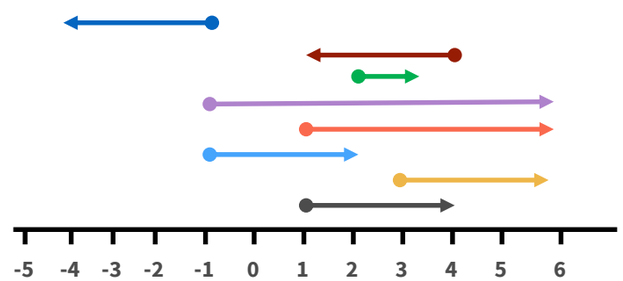
TEST FOR OVERLAPS

- overlap methods



MANIPULATE NUMBER LINE OBJECTS

```
nl <- number_line(1, 4)
shift_number_line(nl,2)
shift_number_line(nl,-2)
expand_number_line(nl, 2, "end")
expand_number_line(nl, 2)
expand_number_line(nl, -1)
reverse_number_line(nl)
invert_number_line(nl)
```



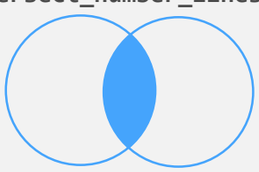
STRUCTURE OF NUMBER LINE OBJECTS

```
> nl <- number_line(1,4); nl
[1] "1 -> 4"
> number_line_width(nl)
[1] 3
> number_line_sequence(nl, 1)
[1] 4 5 6 7
> number_line_sequence(nl, .5)
[1] 4.0 4.5 5.0 5.5 6.0 6.5 7.0
> left_point(reverse_number_line(nl))
[1] 7
> start_point(reverse_number_line(nl))
[1] 4
> right_point(reverse_number_line(nl))
[1] 4
> end_point(reverse_number_line(nl))
[1] 7
```

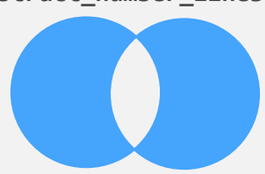
SET OPERATIONS ON NUMBER LINE OBJECTS

union_number_lines()

intersect_number_lines()



subtract_number_lines()



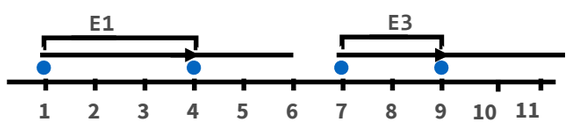
Multi-stage deterministic linkages and case definitions with diyar: : **CHEAT SHEET**

Episode grouping

FIXED EPISODES FROM POINTS IN TIME

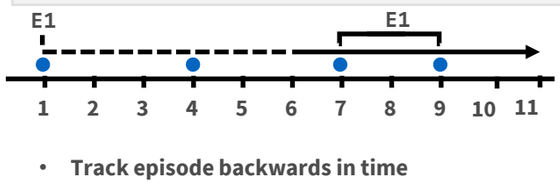
```
dates <- c("01","04","07","09")
dates <- paste(dates,"04/2019",sep= "/")
dates <- as.Date(dates, "%d/%m/%Y")

fixed_episodes(dates, case_length = 5)
```

- A number of days after the index event



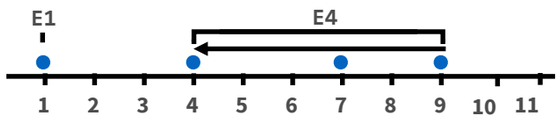
- A range of days after the index event

```
fixed_episodes(dates,
case_length = number_line(5, 10))
```



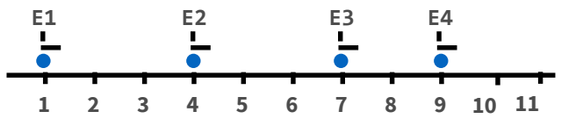
- Track episode backwards in time

```
fixed_episodes(dates, case_length = 5,
from_last = T)
```



- Track episode in other units of time

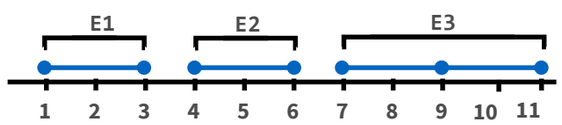
```
fixed_episodes(dates, case_length = 5,
episode_unit = "hours")
```



FIXED EPISODES FROM PERIODS IN TIME

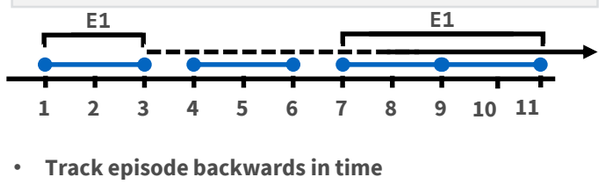
```
p <- as.number_line(dates)
p <- expand_number_line(p,2, "end")
periods <- p

fixed_episodes(periods,case_length = 0)
```

- A number of days after the index period



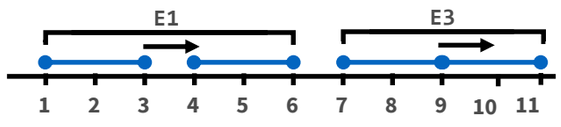
- A range of days after the index period

```
fixed_episodes(periods,
case_length = number_line(5,10)
```



- Track episode backwards in time

```
fixed_episodes(periods, case_length = 1)
```



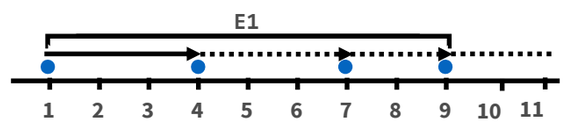
solid lines – case period
dotted lines – recurrence period
dashed lines – skipped period
solid end – start of an episode
arrow head – end of an episode

*Learn more **here!***

ROLLING EPISODES FROM POINTS IN TIME

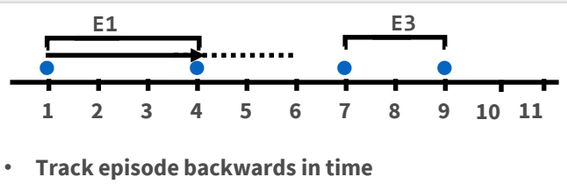
```
rolling_episodes(dates, case_length = 5)
```

- Track episode from events that continue to recur



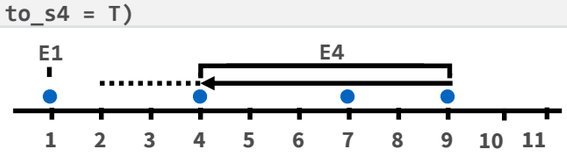
- Track episode from events with a short period of recurrence

```
rolling_episodes(dates, case_length = 5,
recurrence_length =2)
```



- Track episode backwards in time

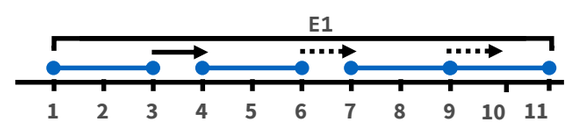
```
rolling_episodes(dates, case_length = 5,
recurrence_length =2, from_last = T,
to_s4 = T)
```



ROLLING EPISODES FROM POINTS IN TIME

```
rolling_episodes(periods, case_length = 1)
```

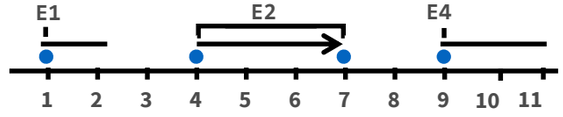
- Track episode from periods that continue to recur



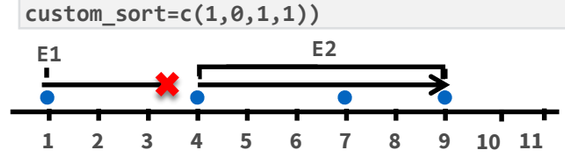
CONTROL CASE ASSIGNMENT

```
fixed_episodes(dates,
case_length = c(1,3,2,2))
```

- Choose your own index event




```
fixed_episodes(dates, case_length =5,
custom_sort=c(1,0,1,1))
```

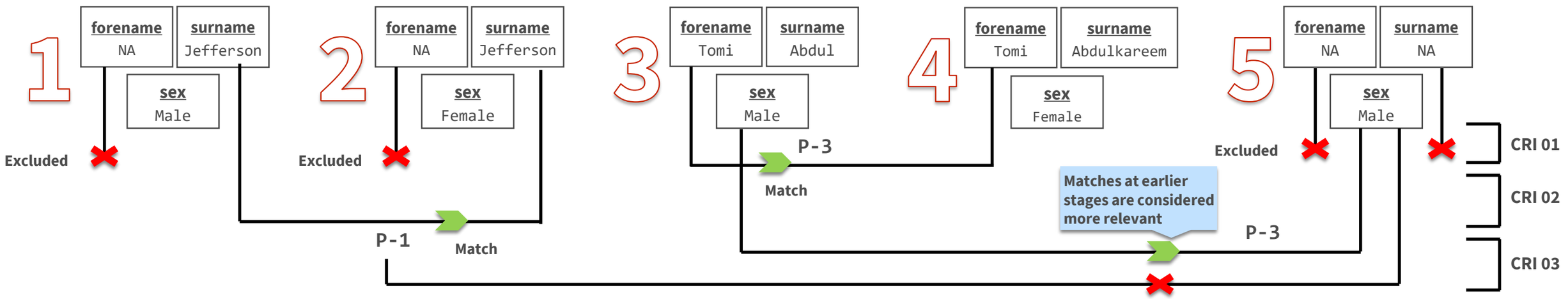


Multi-stage deterministic linkages and case definitions with diyar: : **CHEAT SHEET**

Record grouping

- **Multi-stage deterministic linkage**
- **Relevance of each stage controlled by** `criteria`
- **Use** `sub_criteria` **for additional matching conditions**
- **Missing data handled with alternative matching** `criteria`
- **Group records separately within subsets of a dataset with** `strata`

```
record_group(diyar::patient_list_2, sn = rd_id, criteria = c(forename, surname, sex))
```




```
df <- Opes[c("department","date_of_birth","db_pt1","db_pt2","db_pt3")]
df$age <- dmy("02/11/2019") -  dmy(df$date_of_birth)
df$age <- round(as.numeric(df$age)/365)
```

```
df$age_range <- number_line(df$age, df$age +5, gid=df$age)
record_group(df, criteria = c(department, department),
sub_criteria = list("s1"=c("age_range"), "s2"=c("db_pt1","db_pt2","db_pt3")),
display = FALSE)
```