Contents

1 Data

1.1 Generate

Generate data

```
(defparameter *bigdata* nil)

(with-open-file (stream (merge-pathnames "bigdata-nostrut.csv" *img-path*) :direction
   :if-exists :supersede)
  (let (bigdata)
    (loop for h from 3 to 4 by 0.5 do
      (loop for n from 1 to 16 by 1 do
(loop for bw from 3 to 6 by 0.5 do
  (loop for bx from 2 to 5 do
    (loop for by from bx to 5
  for exact = (simple-building-ftp* :number-of-storey n
    :height h
    :bays-x bx
    :bays-y by
```

```
    :bay-width bw
    :bare nil
    :strut nil)
  do
     (cl-csv:write-csv-row (list n h bw bx by exact)
   :stream stream)
     (push (list (list n h bw bx by) exact) bigdata))))))
    (setf *bigdata* bigdata)))
```

1.2 Read

```
     (let (bigdata)
       (cl-csv:read-csv (merge-pathnames "bigdata-nostrut.csv" *img-path*)
:data-map-fn #'(lambda (data &key csv-reader)
 (declare (ignore csv-reader))
 (let (( *read-default-float-format* 'double-float))
   (read-from-string data)))
:row-fn #'(lambda (row)
    (push (list (butlast row) (first (last row)))
  bigdata)))
       (setf *bigdata* (reverse bigdata)))

nil

NIL
```

2 Simple Fit

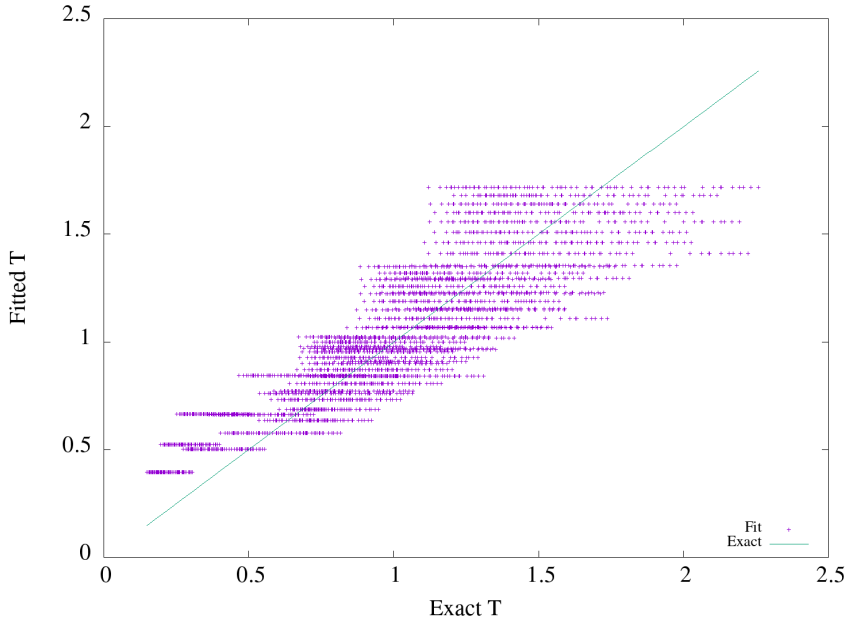
Simpler: $T = a(Nh)^b$ is the best fit among the following 3.

2.1 $T = a*N^b*h^c$

```
(max-fit-%diff (lambda (x a b c)
 (destructuring-bind (n h bw bx by) x
   (* a
      (expt n b)
      (expt h c))))
       '(1 1 1)
       :save '/simple-fit)
```

max %diff	R^2	rmse	Params	Covariance
62.741959	0.735240	0.203328	(0.054824 0.341646 1.801043)	(0.000116 0.000747 0.020485)

```
(scatter-plot /simple-fit "nssimple-fit.png")
```

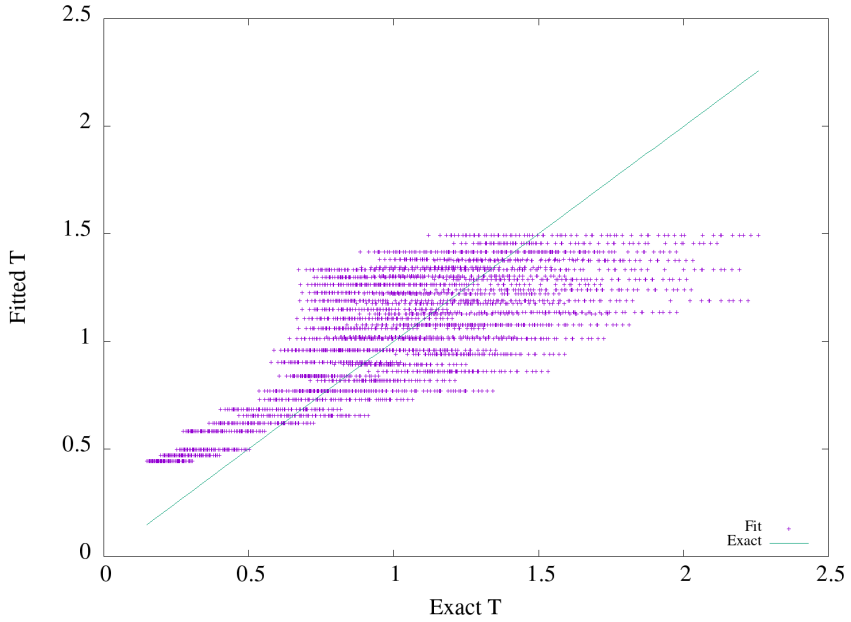


2.2 Simpler: $T = a(Nh)^b$

```
(max-fit-%diff (lambda (x a b)
 (destructuring-bind (n h bw bx by) x
   (* a
      (expt (* n h) b)
      )))
       '(1 1)
       :save '/simpler-fit)
```

max %diff	R^2	rmse	Params	Covariance
87.080565	0.533850	0.269755	(0.286131 0.397355)	(0.000845 0.000789)

```
(scatter-plot /simpler-fit "nssimpler-fit.png")
```



2.3 Simplest: $\mathbf{T = a(N)^b}$

```
(max-fit-%diff (lambda (x a b)
 (destructuring-bind (n h bw bx by) x
   (* a
      (expt (* n) b)
      )))
       '(1 1)
       :save '/simplest-fit)
```

max %diff	R^2	rmse	Params	Covariance
98.557321	0.413727	0.302521	(0.528635 0.341587)	(0.001168 0.000779)

3 Sensitivity Analysis

with Nh only, the variation at higher T is very substantial. Hence other parameters must also be playing a great role in T.

Is it bw or bx,by that play greater role.

Sensitivity of bw is max for h=4,bx=2,by=2. and descreases with decreasing n.

```
(sensitivity :bw (filter-data :n 10 :h 4 :bx 2 :by 2))
```

```
0\.6745613764115661d0
```

Sensitivity with bx doesn't change much with bw.

```
(sensitivity :bx (filter-data :bw 6 :n 10 :h 4))
```

```
0\.1227142212637009d0
```

Hence timeperiod is more sensitive to bay width than with bay counts

3.1 with respect to N, h

```
(sensitivity :n (filter-data :h 4 :bw 3 :bx 5 :by 5))
```

```
0\.8923612826674855d0
```

```
(sensitivity :h (filter-data :n 16 :bw 3 :bx 2 :by 2))
```

```
1\.207306215236251d0
```

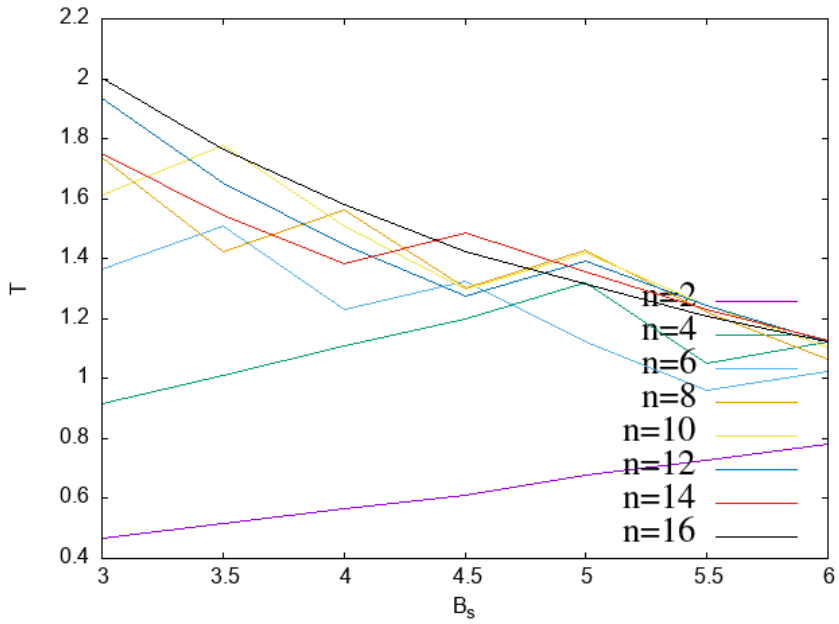
3.2 Max Variations

```
(table 4
  (row "N" "H" "Bs" "Bx" "By" "N and H" "Bx and By")
  (let ((results (loop for p in '(:n :h :bw :bx :by :nh :bxby)
       collect (multiple-value-list (max-effect p)))))
    (apply #'row (mapcar #'first results))
    (apply #'row (mapcar #'second results))))
```

4 Bay width

To incorporate bay width in the relation for fundamental time period. Lets see the variation of T with bw.

```
(with-plot "nsbw.png"
  (setup :xlabel "B_s" :ylabel "T")
  (loop for n from 2 to 16 by 2 do
    (ezplot (mapcar (lambda (d)
      (destructuring-bind (x tp) d
(list (third x) tp)))
    (filter-data :n n :bx 2 :by 2 :h 4))
    :title (format nil "n=~d" n))))
```



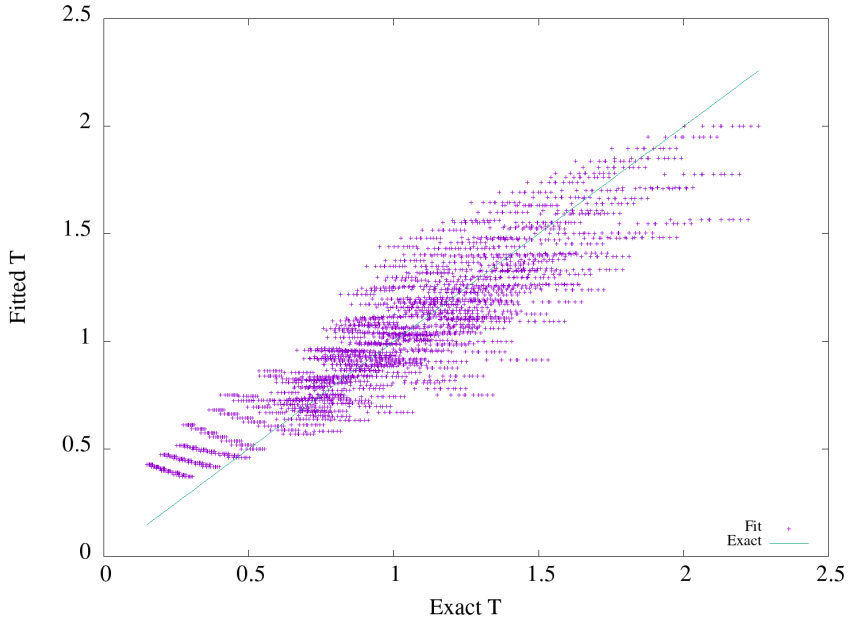
This shows that the time period decreases with bay width. and the rate of decrease/slope increases for higher N.

4.1 Simple Bw a(Nh)^b - c(N*bw)

```
(max-fit-%diff (lambda (x a b c)
 (destructuring-bind (n h bw bx by) x
   (- (* a
 (expt (* n h) b))
      (* c n bw)
 )))
       '(1 1 1)
       :save '/simpler-bw-fit)
```

max %diff	R^2	rmse	Params	Covariance
78.575462	0.798103	0.177556	(0.254081 0.585373 0.018715)	(0.000317 0.000467 0.000003)

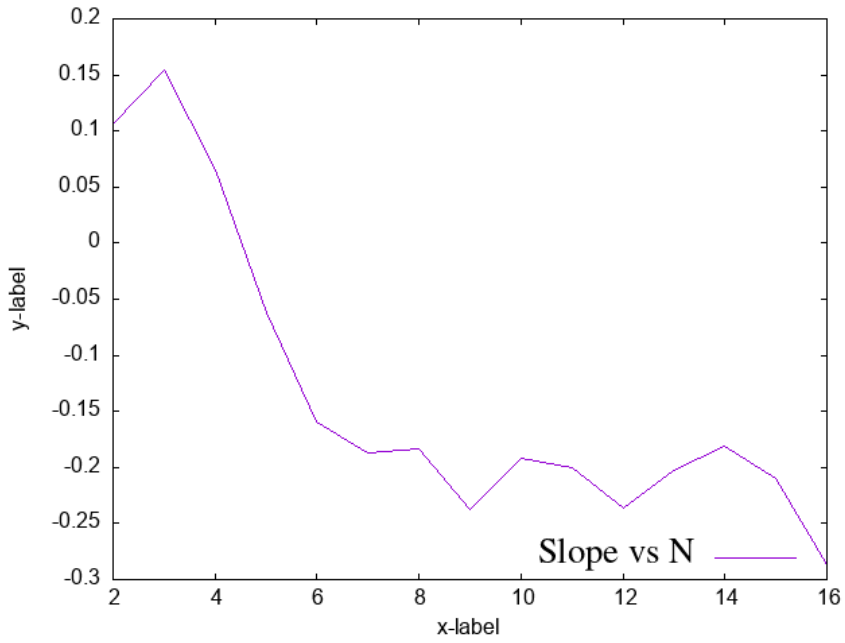
```
(scatter-plot /simpler-bw-fit "nssimpler-bw-fit.png"
      (filter-data ))
```



4.2 A more detailed analysis on variation wrt `bw`

```
(table 4
  (row "N" "Slope" "C" "R^2")
  (loop for n from 2 to 16 by 1
for data = (mapcar (lambda (d)
     (destructuring-bind (x tp) d
       (list (third x) tp)))
   (filter-data :n n :bx 2 :by 2 :h 4))
do
   (multiple-value-bind (_ slope c r) (linear-fit :data data)
     (row n slope c r)))
  (plot-table :file "nsSlope-vs-N.png" :title "Slope vs N"))
```

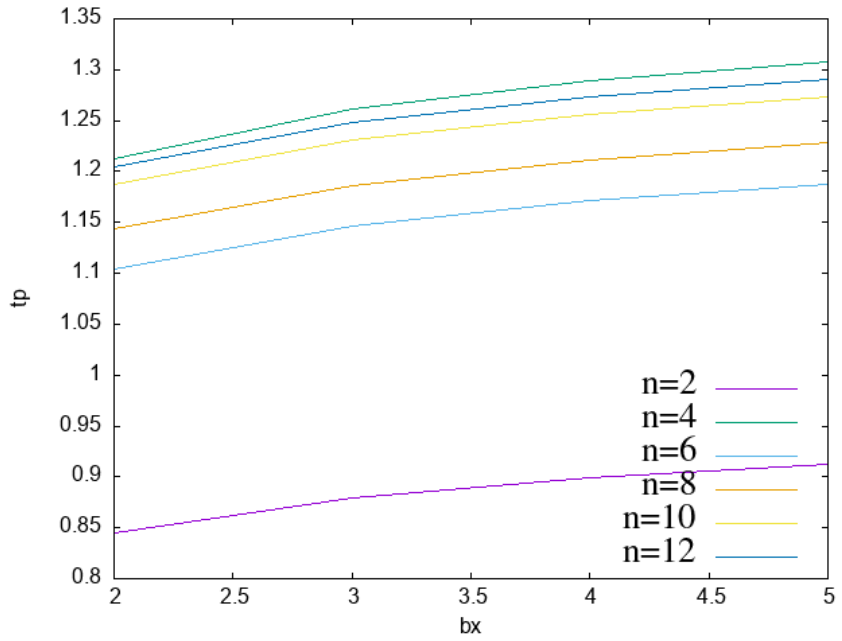
N	Slope	C	R^2
2	0.1062	0.1433	0.9987
3	0.1548	0.2195	0.9990
4	0.0655	0.8089	0.2880
5	-0.0615	1.4467	0.2756
6	-0.1590	1.9351	0.7651
7	-0.1869	2.0780	0.8384
8	-0.1835	2.2169	0.7938
9	-0.2373	2.4925	0.8126
10	-0.1918	2.2871	0.8070
11	-0.2008	2.3335	0.8490
12	-0.2364	2.5022	0.8519
13	-0.2025	2.3374	0.8272
14	-0.1808	2.2247	0.8890
15	-0.2094	2.3853	0.8478
16	-0.2869	2.7787	0.9679



Hence the slope shows almost linear variation with Number of Storeys. Hence, Slope = c * N + d is appropriate which implies correction = - (c * N + d) * bw and finally T = $a(Nh)^b$ - c * N * bw (as in Simple Bw $a(Nh)^b$ - c(N*bw))

5 Bx,By

```
(with-plot "nsbx.png"
  (setup :xlabel "bx" :ylabel "tp")
  (loop for n from 2 to 12 by 2 do
    (ezplot (mapcar (lambda (d)
      (destructuring-bind (x tp) d
(list (fourth x) tp)))
    (filter-data :n n :bw 6 :by 5 :h 4))
    :title (format nil "n=~d" n))))
```



This shows that T is almost constant with bx. Which was also indicated by smaller sensitivity of tp with bx.