Manual for Package: tide

Karl Kastner

July 5, 2019

Contents

1 @T_Tide

1.1 T_Tide

```
wrapper for TPXO generated tidal time series
```

1.2 build_index

```
build a structure whose field names contain the index
```

1.3 from_tpxo

```
read TPXO output into tidetable object
```

1.4 get_constituents

```
extract constituents of tpxo object
```

1.5 reorder

```
order constituents as specified by "name"
```

1.6 select

```
select a subsect of constituents
```

1.7 shift_time_zone

```
shift phase according to time zone
```

2 @Tidal_Envelope

2.1 Tidal_Envelope

```
process tidal data to extrac the tidal envelope
```

2.2 init

```
initialize with data
```

3 @Tide_wft

3.1 Tide_wft

```
wavelet transform of tidal time series
```

3.2 transform

```
wavelet transform tidal time series
input:
time   : [1xn] abszissa of input vector, for example time, must be
    equally spaced
val    : [1xn] signal, input data series (e.g water level or
    velocity)
F      : [1xm] base frequencies, 1, 1, 2, ... for mean level,
    diurnal, semidirunal ...
              base periods from base frequencies T=1/F
n      : [1xm] wavelet window length in multiple of periods
fc, nc : [scalar] low frequency cutoff and window length in periods
winstr : [char] fourier windows (kaiser (recommended), hanning, box
    , etc)
dt_max : [scalar] maximum time to fill gaps in input data series (
    recommended 3/24 for tide)
output:
tide   : struct with fields
        w_coeff  : [1xn] wavelet coefficients (complex)
        amplitude : amplitude
        phase    : phase
        range    :
        h_tide   :
        h_low    :
        h
```

4 @Tidetable

4.1 Tidetable

```
Tide table
```

4.2 analyze

```
extract tidal envelope from time series
```

4.3 export_csv

```
export tide table to csv file
```

4.4 generate

```
run TPXO to generate time series
```

4.5 generate_tpxo_input

```
generate tpxo input table
Note: superseeded by perl script
```

4.6 import_tpxo

```
import TPXO data into tidetable object
```

4.7 plot_neap_spring

```
plot average neap and spring tide
```

5

5.1 constituents

5.2 doodson

```
frequency of tidal constituents
method of doodson
source: wikipedia
```

5.3 envelope_amplitude

```
compute envelopes of hw and low water
```

5.4 envelope_slack_water

```
slack water envelope of the tide
```

5.5 interval_extrema

```
times and evelations for high and low water
```

5.6 interval_extrema2

```
mimimum and maximum within intervals of constant length,
intended for periodic functions
```

5.7 interval_zeros

```
times of slack water determined frim velocity u
```

5.8 lunar_phase

```
lunar phase
```

5.9 rayleigh_criterion

```
raleigh criterion for resolving tidal constituents
T > 1/|f1-f2|
```

6 river-tide/@River_Tide

6.1 River_Tide

```
river tide in a single 1D channel
```

6.2 bcfun

```
boundary conditions
```

6.3 decompose

```
decompose the tide into a right and left travelling wave,
i.e. into incoming and reflected wave
```

6.4 friction_coefficient_dronkers

```
friction coefficient
```

6.5 friction_coefficient_godin

```
friction coefficient of Godin
```

6.6 friction_coefficient_lorentz

```
Loren't friction coefficient
c.f. cai
c.f. dronkers
```

6.7 friction_dronkers

```
friction determined by Dronker's method
```

6.8 friction_exponential_dronkers

```
friction computed by dronkers method
```

6.9 friction_godin

```
compute friction by the method of Godin
```

6.10 friction_trigonometric_dronkers

```
friction computed by dronkers method
expressed as Fourier series coefficients
c.f. dronkers 1964 eq 8.2 and 8.4
Note: Cai dennominates alpha as phi
```

6.11 friction_trigonometric_godin

```
friction computed by the method of godin
expressed in fourier series coefficients
```

6.12 friction_trigonometric_lorentz

```
friction computed by the method of Lorent'z
expressed as Fourier series coefficients
```

6.13 init

```
solve backwater equation for surface level
TODO this should not be solved as a ivp but included in the bvp
    iteration
TODO generate the mesh here and precompute fixed values instead of
    passing functions
```

6.14 mwl_offset

```
offset of the tidally averaged surface elevation caused by tidal
    friction
Linear estimate of the mean water level offset (ignoring feed-back
    of tide)
```

6.15 odefun

```
coefficients of the backwater and wave equation for river-tides
```

6.16 odefun0

```
coefficients of the backwater equation for the river tide
```

6.17 odefun1

```
differential equation coefficients of the main tidal species
f1 Q'' + f2 Q' + f3 Q + f4 = 0
```

6.18 odefun2

```
ordinary differential quation coefficients of the even overtide
```

6.19 q2z

```
tidal component of surface elevation determined from tidal
    discharge

by continuity

%   dz/dt + dq/dx = 0
=> i o z = - dq/dx
=>     z = -1/(io) dq/dx
=>     z = 1i/o dq/dx

TODO allow Q as input
```

6.20 solve

```
call stationary or non-stationary solver respectively
```

6.21 solve_swe

```
determine river tide by the fully non-stationary FVM and then
    extract the tide
```

6.22 solve_wave

```
solve for the oscillatory (tidal) componets
```

6.23 wave_number_analytic

```
analytic expression of the wave number

valid for both tidally, river dominated and low friction conditions
and converging channels

k      : complex wave number in a reach with constant width and bed
    slope
im(k) : damping modulus (rate of amplitude change)
re(k) : actual wave number (rate of phase change)

c.f. derive_wave_number
```

6.24 wave_number_approximation

```
approximate wave number of the left and right traveling wave for
    variable coefficients
```

7 river-tide/@River_Tide_Cai

7.1 Gamma

```
Gamma parameter for tidal propagation
c.f. Cai 2014
```

7.2 River_Tide_Cai

```
prediction of river tide by the method of Cai (2014)
```

7.3 river_tide_cai_

```
determine the surface amplitude of the river tide
c.f. Cai
```

7.4 rt_quantities

```
determine the quantities that determine the tidal propagation
c.f. Cai

Note: this computes 4 unknowns following Cai, however,
lambda, mu and epsilon can be substituted
making it an equation in one unknown (delta) only
```

8 river-tide/@River_Tide_Empirical

8.1 River_Tide_Empirical

```
class for fitting models to at-a-station time series of tidal
    elevation
```

8.2 fit_amplitude

```
fit the oscillatory components
```

8.3 fit_mwl

```
fit the tidally averaged water level
```

8.4 fit_phase

```
fit the phase of the oscillatory components
```

8.5 fit_range

```
fit the tidal range
```

8.6 predict_amplitude

```
predict the oscillatory components
```

8.7 predict_mwl

```
predict the mean water level
```

8.8 predict_phase

```
predict tidal phase
```

8.9 predict_range

```
predict the tidal range
```

8.10 rt_model

```
select the model for fitting
```

9 river-tide/@River_Tide_Map

9.1 River_Tide_Map

```
container class to store individual river tide scenarios
```

9.2 d2au1_dx2

```
second derivative of the tidal velocity magnitude

note: this is for finding zeros,
      the true derivative has to be scaled up by z
```

9.3 d2az1_dx2

```
second derivative of the tidal surface elevation

note: this is for finding zeros,
      the true derivative has to be scaled up by z
```

9.4 dkq_dx

```
along-channel derivative of the wave number of the discharge
neglects width variation

TODO, rederive with g as variable
```

9.5 dkz_dx

```
along channel derivative of the wave number of the tidal surface
    elevation
ignores width variation dh/dx and second order depth variation (d^2
    h/dx^2)
TODO rederive with g symbolic
```

9.6 fun

```
compute a specific river tide scenario and store it in the hash,
or retrive the scenario, if it was already computed
```

9.7 key

```
key for storing a scenario
```

9.8 plot

```
plot result
```

10 river-tide/@Tidal_River_Network

10.1 Tidal_River_Network

```
tide in a fluvial delta channel network, extension of 1D river tide
the network is a directed graph
```

10.2 discharge_amplitude

```
discharge amplitude
```

10.3 mean_water_level

```
predict the mean water level
```

10.4 plot_mean_water_level

```
plot tidally averaged water level
```

10.5 plot_water_level_amplitude

```
plot surface elevation amplitude
```

10.6 solve

```
solve for the tide in a fluvial chanel network

boundary condition at end points not connected to junctions
       [ channel 1 id, endpoint id (1 or 2), s0, c0
       ...
        channel n id, endpoint id (1 or 2), s0, c0]

conditions at junctions are specified as cells
        each cell contains an nx2 array
        n : number of connecting channels
        [channel id1, endpoint id (1 or 2), ...
        channel idn, endpoint id (1 or 2)]

every tidal species for each channel has 4 unknowns
these are 2x2 unknowns for the sin + cos of left and right going
    wave
```

10.7 water_level_amplitude

```
predict the surface elevation amplitude
```

11 river-tide/River_Tide_JK

11.1 River_Tide_JK

11.2 jk_damping_modulus

```
damping modulus of the river tide
c.f. Jay and Kukula
```

11.3 jk_mean_level

```
tidally averaged surface elevation
c.f. Jay and Kukulka
```

11.4 jk_rivertide_predict

```
predict river tide by the method of jay and kukulka
```

11.5 jk_rivertide_regress

```
Regression of tidal coefficients according to Jay & Kulkulka

coefficients of the r-regression factor 2 apart for specis (jay C7)
this can be repeated for each tidal species (diurnal, semidiurnal)
```

11.6 jk_tidal_discharge

```
tidal discharge
c.f. Jay and Kukulka
```

11.7 jk_tidal_range

```
predict tidal range
```

12 river-tide

analysis and prediction of river tides

12.1 damped_wave_bvp

```
analytic solution to the river tide formulated as boundary value
    problem
in a river with finite length

c.f. Godin 1986
```

12.2 damped_wave_ivp

```
linearly damped wave in rectangular channel
x_t = Ax + b
```

12.3 damping_modulus_river

```
damping modulus of the tidal wave for river flow only
```

12.4 damping_modulus_tide

```
damping modulus of the tide without river flow
c.f. friedrichs, ippen harleman
output :
k : wave number
re(k) : rate of phase change
im(k) : damping rate
```

12.5 rdamping_to_cdrag_tide

```
converts damping rate to drag coefficient
c.f. friedrichs, ippen harleman
```

12.6 rt_celerity

```
celerity of the tidal wave
```

12.7 rt_quasi_stationary_complex

```
quasi-stationary solution of the SWE
TODO staggered grid does not help: q1' needed
```

12.8 rt_quasi_stationary_trigonometric

```
quasi statinary form of the SWE
```

12.9 rt_reflection_coefficient_gradual

```
reflection coefficient for gradual varying cross section geometry
without damping
```

12.10 rt_wave_equation

```
solve river tide as boundary value problem

input:
omega : [nfx1] angluar frequency of tidal component, zero for mean
    flow
reach : [nrx1] struct
.L    : [1x1] length of reaches
        .width(x,h)   width
        .bed(x,h)     bed level
        .surface(x,h) surface elevation
        .Cd(x,h)      drag coefficient
.bc   : [nd,nf] boundary/junction conditions
        bc(id,if).type : {surface, velocity, discharge} (dirichlet)
        bc(id,if).val : value
opt   : [1x1] struct
       - constant surface elevation
       - deactivative advective acceleration
       .dx : spatial resolution

dimensions:
       nr : nurmber or reaches
       nd : upstream/downstream index
       nf : frequency index
```

12.11 rt_z2q

```
determine tidal discharge from water level for tidal wave
```

12.12 test_rt_wave_number

12.13 tidal_ellipse

```
tidal ellipse, numerical ode solution
```

12.14 wavetrainz

```
determine river tide by iterated integration of the surface
    elevation
```

12.15 wavetwopassz

```
two pass solution for the linearised wave equation, for surface
    elevation
```

13 test

13.1 test_tidal_harmonic_analysis

14

14.1 tidal_constituents

14.2 tidal_energy_transport_1d

```
energy transport of a tidal wave
```

14.3 tidal_envelope

```
envelope of the tide

input : t time in days
       f surface elevation
ouput : tl time of low water
       vl surface elevation at low water
       ldx index of low water
       th time of high water
       vh surface elevation at high water
       hdx index of high water
       ndx neap index
       sdx spring index
       dmax:
       drange: range per day
```

14.4 tidal_envelope2

```
surface levelation envelope of the tide
low water, high water and tidal range for lunar each day


input:
      time  :
      L     : surface elevation
      order : interpolation order (default 2)
ouput:
      timei : vector eqispaced
      lmini : minimum level
      lmaxi : maximum level
      rangei : range
      midrangei : (min + max)/2, usually different from mean
      phii  : pseudo phase

Note: the pseudo phase phi jumps, this is because if the tide is
    semidiurnal,
      sometimes the lower hw becomes the next day higher then than
          the
      current high water, e.g. there is no smooth transition by
```

```
    51min but a jump by 12h
```

14.5 tidal_harmonic_analysis

```
tidal_harmonic analysis
```

15 tide-savenije

15.1 savenije_phase_lag

```
phase lag of high and low water

phi : u_river/u_tide < 1

delta_eps_hw = omega*(t_hws - t_hw)
delta_eps_hw = omega*(t_lws - t_lw)

c.f. savenije
```

15.2 savenije_tidal_range

```
tidal range

based on Savenije 2012

x    : distance to river mouth
eta  : range
eta0 : range at river mouth
hbar : mean water depth
phi  : velocity ratio u_tide/u_river
       note: this varies in strongly convergent estuaries
K    : mannings coefficient
I    : residual surface slope I
```

15.3 savenije_tidal_range1

```
tidal range

based on Horrevoets/Savenije, 2004
```

```
H0   : tidal range at river mouth
h0   : initial water depth
v    : velocity scale
b    : convergence length
sine : phase lag
K    : Mannings coefficient
Q_r   : river discharge
```

15.4 savenije_timing_hw_lw

```
time of high water and low water
c.f. savenije 2012
```

15.5 tide-savenije