

Manual for Package: tide

Revision 6:14M

Karl Kästner

April 20, 2020

Contents

1	@T_Tide	1
1.1	T_Tide	1
1.2	build_index	1
1.3	from_tpxo	1
1.4	get_constituents	1
1.5	reorder	1
1.6	select	2
1.7	shift_time_zone	2
2	@Tidal_Envelope	2
2.1	Tidal_Envelope	2
2.2	init	2
3	@Tide_wft	2
3.1	Tide_wft	2
3.2	transform	2
4	@Tidetable	3
4.1	Tidetable	3
4.2	analyze	3
4.3	export_csv	3
4.4	generate	3
4.5	generate_tpxo_input	3
4.6	import_tpxo	4
4.7	plot_neap_spring	4
5	tide	4
5.1	constituents	4
5.2	doodson	4
5.3	envelope_amplitude	4

5.4	envelope_slack_water	4
5.5	interval_extrema	4
5.6	interval_extrema2	5
5.7	interval_zeros	5
5.8	lunar_phase	5
5.9	rayleigh_criterion	5
6	river-tide/@River_Tide	5
6.1	River_Tide	8
6.2	bc_transformation	8
6.3	bcfun	8
6.4	check_continuity	9
6.5	check_momentum	9
6.6	d2au1_dx2	9
6.7	d2az1_dx2	9
6.8	decompose	9
6.9	discharge2level	9
6.10	dkq_dx	10
6.11	dkz_dx	10
6.12	even_overtide_analytic	10
6.13	friction_coefficient	10
6.14	friction_coefficient_dronkers	10
6.15	friction_coefficient_godin	11
6.16	friction_coefficient_lorentz	11
6.17	friction_dronkers	11
6.18	friction_exponential_dronkers	11
6.19	friction_godin	12
6.20	friction_lorentz	12
6.21	friction_quadratic	12
6.22	friction_trigonometric_dronkers	12
6.23	friction_trigonometric_godin	12
6.24	friction_trigonometric_lorentz	12
6.25	generate_delft3d	12
6.26	init	13
6.27	initial_value	13
6.28	mwloffset	13
6.29	mwloffset_2	13
6.30	mwloffset_analytic	13
6.31	odefun	13
6.32	odefun0	13
6.33	odefun_advective_acceleration	14
6.34	odefun_friction	14
6.35	odefun_ghof	14
6.36	odefun_swe_jacobian	14

6.37	odefun_width	14
6.38	odefunk	14
6.39	solve	14
6.40	solve_backwater	15
6.41	solve_swe	15
6.42	solve_wave	15
6.43	wave_number_analytic	15
6.44	wave_number_approximation	15
7	river-tide/@River_Tide_Cai	16
7.1	Gamma	16
7.2	River_Tide_Cai	16
7.3	river_tide_cai_	16
7.4	rt_quantities	16
8	river-tide/@River_Tide_Empirical	16
8.1	River_Tide_Empirical	16
8.2	fit_amplitude	17
8.3	fit_mwl	17
8.4	fit_phase	17
8.5	fit_range	17
8.6	predict_amplitude	17
8.7	predict_mwl	17
8.8	predict_phase	17
8.9	predict_range	17
8.10	rt_model	18
9	river-tide/@River_Tide_JK	18
9.1	River_Tide_JK	18
9.2	damping_modulus	18
9.3	mean_level	18
9.4	rivertide_predict	18
9.5	rivertide_regress	18
9.6	tidal_discharge	19
9.7	tidal_range	19
10	river-tide/@River_Tide_Map	19
10.1	River_Tide_Map	19
10.2	fun	19
10.3	key	19
10.4	plot	19
11	river-tide/@River_Tide_Network	20
11.1	River_Tide_Network	20

11.2	discharge_amplitude	20
11.3	mean_water_level	20
11.4	plot_mean_water_level	20
11.5	plot_water_level_amplitude	20
11.6	solve	20
11.7	water_level_amplitude	21
12	river-tide	21
12.1	damped_wave_bvp	21
12.2	damped_wave_ivp	21
12.3	damping_modulus_river	22
12.4	rdamping_to_cdtag_tide	22
12.5	river_tide_godin	22
12.6	rt_celerity	22
12.7	rt_quasi_stationary_complex	22
12.8	rt_quasi_stationary_trigonometric	22
12.9	rt_reflection_coefficient_gradual	23
12.10	rt_wave_equation	23
12.11	rt_z2q	23
13	river-tide/test	23
13.1	test_bvp2c2	23
13.2	test_bvp2c_sym	24
13.3	test_celerity	24
13.4	test_characteristic_rate_of_change	24
13.5	test_complex_even_overtide	24
13.6	test_dronkers_compound	24
13.7	test_fourier_power_exp	24
13.8	test_friction	24
13.9	test_friction_dronkers	24
13.10	test_friction_dronkers2	24
13.11	test_fv_compare_schemes	24
13.12	test_fv_convergence	25
13.13	test_power_series	25
13.14	test_reflection	25
13.15	test_reflection_coefficient_gradual	25
13.16	test_ricatti	25
13.17	test_river_tide_models	25
13.18	test_rt_reflection	25
13.19	test_rt_wave_number	25
13.20	test_rt_zs0	25
13.21	test_swe	25
13.22	test_tidal_river_network	26
13.23	test_tidal_river_network_z0	26

13.24	test_tide_slack_exp	26
13.25	test_utm2latlon	26
13.26	test_wave_number_godin	26
13.27	test_wave_numer_aproximation	26
13.28	test_wave_twopass	26
14	river-tide	26
14.1	tidal_ellipse	27
14.2	tide_slack_exp	27
14.3	wave_number_tide	27
14.4	wavetrainz	27
14.5	wavetwopassz	27
15	test/river-tide	28
15.1	example_river_tide	28
15.2	example_river_tide_map	28
15.3	river_tide_test_01	28
15.4	river_tide_test_02	28
15.5	river_tide_test_03	28
15.6	river_tide_test_04	28
15.7	river_tide_test_05	28
15.8	river_tide_test_06	28
15.9	river_tide_test_07	28
15.10	river_tide_test_08	29
15.11	river_tide_test_09	29
15.12	river_tide_test_10	29
15.13	river_tide_test_11	29
15.14	river_tide_test_12	29
15.15	river_tide_test_13	29
15.16	river_tide_test_batch	29
15.17	river_tide_test_metadata	29
15.18	river_tide_test_plot	29
16	test	30
16.1	test_tidal_harmonic_analysis	30
17	tide	30
17.1	tidal_constituents	30
17.2	tidal_energy_transport_1d	30
17.3	tidal_envelope	30
17.4	tidal_envelope2	30
17.5	tidal_harmonic_analysis	31
17.6	tidal_range_exp	31
17.7	tidal_range_tri	31

18	tide-savenije	31
18.1	savenije_phase_lag	31
18.2	savenije_tidal_range	32
18.3	savenije_tidal_rangel	32
18.4	savenije_timing_hw_lw	32
18.5	tide-savenije	32
19	tide	33
19.1	tide_low_high_exp	33
19.2	tide_low_high_tri	33

1 @T_Tide

1.1 T_Tide

wrapper for TPX0 generated tidal time series

1.2 build_index

build a structure whose field names contain the index

1.3 from_tpxo

read TPX0 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 subset 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

wavelet analysis of tidal data

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

class for generating tidetable data

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: superseded by perl script

4.6 import_tpxo

import TPX0 data into tidetable object

4.7 plot_neap_spring

plot average neap and spring tide

5 tide

analysis prediction of tides in rivers and estuaries by empirical
and theoretical methods

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 elevations for high and low water

5.6 interval_extrema2

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

5.7 interval_zeros

times of slack water determined from velocity u

5.8 lunar_phase

lunar phase

5.9 rayleigh_criterion

rayleigh criterion for resolving tidal constituents
 $T > 1/|f_1 - f_2|$

6 river-tide/@River_Tide

predict tide in a backwater affected river with a sloping/varying
bed

Assumptions and capabilities:

- tidal dynamics follow the 1D-Shallow-Water-Equation
(depth and cross-sectionally averaged Navier-Stokes-
Equation)
- rectangular cross section
- width can vary along the channel
- friction coefficient (cd) constant along channel and over
time (Chezy)
- advective acceleration term is considered, but can be
deactivated
- vertical profile of streamwise velocity is constant
(Boussinesq coefficient is unity (1))

Limitations / TODO list:

- single channel dynamics only (no tidal networks)
- no wind-shear stress (no storm surges)
- no tidal flats / intertidal areas (width constant in time)

- no flood-plain during high-river flow
 - no stratification or along-channel salinity gradient
 - negligible head loss in channel bends
 - negligible feed-back of the sediment concentration on the propagation of the tide
 - low Froude Number (no hydraulic jumps due to cataracts or tidal bores)
- At present, only two tidal components are supported (either D1 with D2 or D2 with D4, in addition to the mean water level z_0), for mixed diurnal-semidiurnal cases with dominant semidiurnal component, the class has to be extended to support three components (D1, D2 and D4)
 - At present, the tripel overtide is not computed (D3 for diurnal, D6 for semidiurnal tide), note that this is the main overtide for the case of low river flow
 - At present, the $1/h$ non-linearity is only included in the approximations of the backwater curve, but not it's influence on the tidal frequency components

Method:

This class calls numerical solvers for second order ordinary differential equation boundary value problems

Tides is represented as exponential series in form of total discharge $Q = \sum Q_i = Q_0 + Q_1 + Q_2$, as discharge is conserved (balanced), the equations are simpler than for level z and velocity u , and the frequency components of z are straight forward determined by differentiation of Q

Class and function structure:

```

River_Tide :
    computes river tide, provides the ode coefficients to
    the boundary value solver
bvp2c, bvp2fdm :
    solve the underlying second order boundary value
    problem
River_Tide_Map :
    provides convenient batch runs and processing of
    River_Tide instances

```

Minimum working example, c.f. example_rive_tide.m and
example_river_tide_map.m

input:

Q0 : scalar, river discharge (m^3/s)
omega : scalar, angular frequency main tidal species in (1/seconds)
x : 2x1 vector, left and right end of computational domain of the river (m)
w(x) : function of width along the river (m)
cd(x) : function of drag coefficient along the river (1)
zb(x) : function of bed level along the river (m)

opt : structure with options
opt.model_str = 'wave' (other solver are not supported at the moment)
opt.solver = @bvp2c or @bvp2fdm
opt.nx : number of grid points along channel
opt.ns : base for logarithmic spacing of grid points, 1 : linear spacing

bc : structure array of boundary conditions
r, row 1..2 : left and right end, respectively
c, column 1 : mean (river) component
2..n : condition form column-1 frequency component

$$\begin{pmatrix} q(1)*(p(1) \hat{y}^-(x_0) + p(2) \frac{d\hat{y}^-}{dx}(x_0) \dots \\ + q(2)*(p(1) \hat{y}^+(x_0) + p(2) \frac{d\hat{y}^+}{dx}(x_0)) = bc(c, r).val \\ = val(0) \end{pmatrix}$$

bc(c,r).var : Quantity, either 'z' or 'Q'
bc(c,r).val : complex amplitude of chosen variable
(c.f. (1 + 0i) [m] for surface elevation amplitude of 1m)
(value has to be real for mean component)
mean component requires z and Q to be specified at opposit ends

bc(c,r).p : factor for Dirichlet p(1) or Neumann p(2) condition
p = [1,0] : pure Dirichlet
p = [0,1] : pure Neumann
sum of abs(p) must be nonzero for each end and each frequency component

bc(c,r).q : factor for left and right going wave, only available for bvp2c
q = [1,1] : total water level / discharge

```

q = [1,0] : only left going wave
q = [0,1] : only right going wave
q has no meaning for the mean component and is
    ignored
q is only supported by bvp2c,
bvpfdm uses default q = [1,1]
sum of abs(q) for each frequency component must
    be zero

```

6.1 River_Tide

```

river tide in a single 1D channel
TODO split in two classes:
one that stores data (RT_Solve), one that provides equations (
    RT_Analytic)

```

6.2 bc_transformation

6.3 bcfun

```

Robin (mixed) boundary conditions for the river tide,
supplied for each frequency component,
wrapper that copies values are from the member struct "bc"

```

```

    q*(p*Q_1^- + (1-p)*dQ_1^-/dx
input :
    x      : coordinate (left or right end)
    id,ccdx : frequency component index
              (1 = 0 omega (mean), 2 : 1 omega, 3 : 2 omega, ...
columns of bc : frequency
rows of bc, left, right boundary
output :
    p : [2x1] linear combination of Dirichlet and Neumann
        boundary condition
    p(1) -> weight Dirichlet boundary condition
    p(2) -> weight Neumann boundary condition
    q linear combination of left and right travelling (incoming and
        outgoing) wave
    q(1) weight left going wave
    q(2) weight right going wave
    rhs = 0 -> homogeneous boundary condition

```

```

function [rhs, p, q, obj] = bcfun(obj,x,y,ccdx)

```

6.4 check_continuity

6.5 check_momentum

6.6 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

6.7 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

6.8 decompose

decompose the tide into a right and left travelling wave,
i.e. into incoming and reflected wave
TODO subtract forcing term

6.9 discharge2level

tidal component of surface elevation determined from tidal
discharge

by continuity :

$$\begin{aligned} dz/dt + dq/dx &= 0 \\ \Rightarrow i \circ z &= - dq/dx \\ \Rightarrow z &= -1/(io) dq/dx \\ \Rightarrow z &= 1i/o dq/dx \end{aligned}$$

TODO allow Q as input
TODO rename into Q1_to_z1
Mon 7 Oct 19:04:14 PST 2019 : added correction for change of width

6.10 dkq_dx

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

TODO, rederive with g as variable

6.11 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^2h/dx^2)
TODO rederive with g symbolic

6.12 even_overtide_analytic

6.13 friction_coefficient

6.14 friction_coefficient_dronkers

friction coefficient according to Dronkers

the coefficients are semi-autogenerated

c.f. dronkers 1964
c.f. Cai 2016

$p = [p_0, p_1, p_2, p_3];$
 $\alpha = U_r/U_t = \text{river velocity} / \text{tidal velocity amplitude} = (u_{\max} + u_{\min}) / (u_{\max} - u_{\min})$

function p = friction_coefficient_dronkers(alpha,order)

6.15 friction_coefficient_godin

friction coefficient according to Godin
these coefficients are identical to Dronker's for $U_R = \phi = 0$

```
function G = friction_coefficient_godin(obj,phi)
```

6.16 friction_coefficient_lorentz

friction coefficient according to Lorentz
identical to Dronker's coefficient for zero river flow
and a single frequency component
c.f. Cai
c.f. Dronkers

```
function L = friction_coefficient_lorentz(obj,phi)
```

6.17 friction_dronkers

friction determined by Dronker's method

input :

- u : velocity time series
- Umid : arithmetic mean of minimum and maximum velocity
(not the mean of the velocity, usually non-zero even
without river flow)
- Uhr : half-range of the velocity, less than the sum of
the frequency amplitudes, except at perigean spring
tides

```
function [uau_sum uau p] = friction_dronkers(u,Umid,Uhr,order)
```

6.18 friction_exponential_dronkers

friction coefficients for the frequency components computed by
Dronkers method
c.f. Dronker's 1964 eq 8.2 and 8.4
Note: Cai denominates alpha as phi

```
function [c uau uau_p] = friction_trigonometric_dronkers(u,dp,Umid,  
    ,Uhr,order,psym)
```


6.19 friction_godin

compute friction with the method of Godin

6.20 friction_lorentz

6.21 friction_quadratic

friction determined by Dronker's method

6.22 friction_trigonometric_dronkers

friction computed by the method of Dronkers
expressed as coefficients for the frequency components
c.f. dronkers 1964 eq 8.2 and 8.4
Note: Cai dennominates alpha as phi

6.23 friction_trigonometric_godin

friction computed by the method of Godin
expressed as coefficients of the frequency components (
trigonometric form)

```
function [c, uau] = friction_trigonometric_godin(obj,u,dp,Umax)
```

6.24 friction_trigonometric_lorentz

friction computed by the method of Lorent's
expressed as coefficients of the frequency components (
trigonometric form)

6.25 generate_delft3d

6.26 init

provide initial condition by solving the 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
TODO Q0 should not be a function
function obj = init(obj, Xi)

6.27 initial_value

6.28 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)

6.29 mwl_offset_2

6.30 mwl_offset_analytic

6.31 odefun

coefficients of the backwater and wave equation for river-tides

6.32 odefun0

coefficients of the backwater equation for the river tide
TODO merge with backwater

6.33 odefun_advective_acceleration

6.34 odefun_friction

6.35 odefun_ghof

6.36 odefun_swe_jacobian

6.37 odefun_width

6.38 odefunk

coefficients of the ordinary differential equation of the k-th
frequency
component of the tide

$$f1 Q'' + f2 Q' + f3 Q + f4 = 0$$

TODO rename f into c

TODO better pass dzb_dx instead of dz0_dx

TODO aa, oh and gh terms are not tested for width ~ 1

6.39 solve

call stationary or non-stationary solver respectively

function obj = solve(obj)

6.40 solve_backwater

6.41 solve_swe

determine river tide by the fully non-stationary FVM and then
extract the tide
this is experimental and not yet fully working

6.42 solve_wave

solve for the oscillatory (tidal) componets
function obj = solve_wave(obj)

6.43 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

6.44 wave_number_approximation

approximate wave number of the left and right traveling wave for
variable coefficients
TODO merge with wave_number_analytic
function [k, k0, dk0_dx_rel, obj] = wave_numer_aproximation(obj)

7 river-tide/@River_Tide_Cai

Prediction of river tide by the method of Cai
c.f. Cai 2013, Cai 2015

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

Empirical fit to measurement and prediction (from tide at sea and
river discharge)
of the river tide

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_JK

empirical analysis and prediction of river tides by the method of
Jay and Kukulka

9.1 River_Tide_JK

9.2 damping_modulus

damping modulus of the river tide
c.f. Jay and Kukulka
function r = damping_modulus(obj,h0,b,Qr)

9.3 mean_level

tidally averaged surface elevation
c.f. Jay and Kukulka

9.4 rivertide_predict

predict river tide by the method of jay and kukulka
TODO rename

9.5 rivertide_regress

Regression of tidal coefficients according to Jay & Kukulka

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

9.6 tidal_discharge

tidal discharge
c.f. Jay and Kukulka
function Qt = tidal_discharge(obj,x,R0,h0,b,Qr)

9.7 tidal_range

predict tidal range

10 river-tide/@River_Tide_Map

hash container for a set of River_Tide predictions for different
boundary conditions

10.1 River_Tide_Map

container class to store individual river tide scenarios

10.2 fun

compute river tide for a scenario with specific boundary conditions
and store it in the hash,
or retrieve the scenario, if it was already computed

10.3 key

key for storing a scenario

function [key obj] = key(obj,varargin)

10.4 plot

quick plot of scenario result

function obj = plot(obj,Xi,Q0,W0,S0,z1_downstream,cd,zb_downstream,
omega,q,opt)

11 river-tide/@River_Tide_Network

predict tides in a fluvial channel network

11.1 River_Tide_Network

tide in a fluvial delta channel network, extension of 1D river tide
the network is a directed graph
TODO convert from trig-to exponential form

11.2 discharge_amplitude

discharge amplitude

11.3 mean_water_level

predict the mean water level

11.4 plot_mean_water_level

plot tidally averaged water level

11.5 plot_water_level_amplitude

plot surface elevation amplitude

11.6 solve

solve for the tide in a fluvial channel 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

11.7 water_level_amplitude

predict the surface elevation amplitude

12 river-tide

analysis and prediction of river tides

Sub-Classes:

```
@River_Tide
    - prediction of river tide in a backwater affected river with
      a sloping bed
@River_Tide_Cai
    - prediction of river tide, method of Cai
@River_Tide_Empirical
    - prediction of river tide, empirical
@River_Tide_JK
    - prediction of river tide, empirical after Jay and Kukulka
@River_Tide_Map
    - mulitple-scenaria container for River_Tide
@River_Tide_Network
    - extension of River_Tide to networks
```

12.1 damped_wave_bvp

solved damped wave equation
 $z'' + a z = 0$
 $z(0) = z_0, z(L) = 0$

12.2 damped_wave_ivp

linearly damped wave in rectangular channel
solve tide as initial value problem
damped wave approximation

$$z'' + a z = 0$$

$$x_t = Ax + b$$

12.3 damping_modulus_river

damping modulus of the tidal wave for river flow only

12.4 rdamping_to_cdrag_tide

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

12.5 river_tide_godin

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

c.f. Godin 1986

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 stationary 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

13 river-tide/test

13.1 test_bvp2c2

13.2 test_bvp2c_sym

13.3 test_celerity

13.4 test_characteristic_rate_of_change

13.5 test_complex_even_overtide

13.6 test_dronkers_compound

13.7 test_fourier_power_exp

13.8 test_friction

13.9 test_friction_dronkers

13.10 test_friction_dronkers2

13.11 test_fv_compare_schemes

13.12 `test_fv_convergence`

13.13 `test_power_series`

13.14 `test_reflection`

13.15 `test_reflection_coefficient_gradual`

13.16 `test_ricatti`

13.17 `test_river_tide_models`

13.18 `test_rt_reflection`

13.19 `test_rt_wave_number`

13.20 `test_rt_zs0`

13.21 `test_swe`

13.22 test_tidal_river_network

13.23 test_tidal_river_network_z0

13.24 test_tide_slack_exp

13.25 test_utm2latlon

13.26 test_wave_number_godin

13.27 test_wave_numer_aproximation

13.28 test_wave_twopass

14 river-tide

analysis and prediction of river tides

Sub-Classes:

@River_Tide

- prediction of river tide in a backwater affected river with a sloping bed

@River_Tide_Cai

- prediction of river tide, method of Cai

@River_Tide_Empirical

- prediction of river tide, empirical

```

@River_Tide_JK
    - prediction of river tide, empirical after Jay and Kukulka
@River_Tide_Map
    - mulitple-scenaria container for River_Tide
@River_Tide_Network
    - extension of River_Tide to networks

```

14.1 tidal_ellipse

tidal ellipse, numerical ode solution

14.2 tide_slack_exp

14.3 wave_number_tide

wave number of the tide without river flow
 c.f. friedrichs, ippen harleman
 output :
 k : wave number, such that

$$z(t,x) = z_1(t,0) \exp(i\omega t - kx)$$

 re(k) : rate of phase change
 -im(k) : damping rate
 function [k k_low k_high] = damping_modulus_tide(omega,cd,h0,az1)

14.4 wavetrainz

determine river tide by iterated integration of the surface elevation

14.5 wavetwopassz

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

15 test/river-tide

15.1 example_river_tide

15.2 example_river_tide_map

15.3 river_tide_test_01

15.4 river_tide_test_02

15.5 river_tide_test_03

15.6 river_tide_test_04

15.7 river_tide_test_05

15.8 river_tide_test_06

15.9 river_tide_test_07

15.10 river_tide_test_08

```
hold on;  
plot(x,abs(z),'--');  
hold on;  
plot(x,angle(z),'--');
```

15.11 river_tide_test_09

15.12 river_tide_test_10

15.13 river_tide_test_11

15.14 river_tide_test_12

15.15 river_tide_test_13

15.16 river_tide_test_batch

15.17 river_tide_test_metadata

15.18 river_tide_test_plot

16 test

16.1 test_tidal_harmonic_analysis

17 tide

analysis prediction of tides in rivers and estuaries by empirical and theoretical methods

17.1 tidal_constituents

17.2 tidal_energy_transport_1d

energy transport of a tidal wave

17.3 tidal_envelope

envelope of the tide

input : t time in days
 f surface elevation
output : 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

17.4 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)
output:
    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 ϕ 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

17.5 tidal_harmonic_analysis

tidal_harmonic analysis

17.6 tidal_range_exp

17.7 tidal_range_tri

18 tide-savenije

18.1 savenije_phase_lag

phase lag of high and low water

$\phi : u_{\text{river}}/u_{\text{tide}} < 1$

$\text{delta_eps_hw} = \omega \cdot (t_{\text{hws}} - t_{\text{hw}})$

$\text{delta_eps_hw} = \omega \cdot (t_{\text{lws}} - t_{\text{lw}})$

c.f. savenije

18.2 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 $u_{\text{tide}}/u_{\text{river}}$
note: this varies in strongly convergent estuaries
K : mannings coefficient
I : residual surface slope I

18.3 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

18.4 savenije_timing_hw_lw

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

18.5 tide-savenije

19 tide

analysis prediction of tides in rivers and estuaries by empirical
and theoretical methods

19.1 tide_low_high_exp

19.2 tide_low_high_tri