Ordinary Differential Integrators

Daniel Celis Garza

April 29, 2016

1 Directory Tree

`./applied_math/find_roots`

2 Dependencies

`nrtype.f08`

3 Subroutines

3.1 Bracketing Root by Expanding an Interval

`outbracket(func, interval, max_it, success, factor)`

3.1.1 Arguments

```
  subroutine outbracket(func, interval, max_it, success, factor)
    !=======================================================!
    ! Gradually expands bracket until it finds a root.      !
    !-------------------------------------------------------!
    ! Cycles through a number of iterations until the       !
    ! maximum number of iterations is reached. The interval !
    ! is expanded according to a user-provided factor. If   !
    ! the cycle finds a root, it exits the subroutine with  !
    ! success = .true., else it exits with success = .false.!
    ! and a greater interval.                               !
    !-------------------------------------------------------!
    ! Inputs:                                               !
    ! func   = function whose roots we're trying to bracket !
    ! factor = optional factor for increasing the interval  !
    ! max_it = maximum number of iterations                 !
    !-------------------------------------------------------!
    ! Outputs:                                              !
    ! success = logical value stating whether a root was    !
    !           bracketted                                  !
    !-------------------------------------------------------!
    ! Inputs-Outputs:                                       !
    ! interval() = input a proposed interval, output an     !
    !              interval which brackets the root.        !
    !-------------------------------------------------------!
    ! Locals:                                               !
```

```
! f() = value of func() at the interval bounds          !
! i   = counter                                         !
!=======================================================!
implicit none
interface cont_func
  function func(x)
    use nrtype
    implicit none
    real(dp), intent(in) :: x
    real(dp)             :: func
  end function func
end interface cont_func
real(dp), intent(inout) :: interval(2)
integer,  intent(in)    :: max_it
logical,  intent(out)   :: success
real(dp), optional      :: factor
real(dp)                :: f(2)
integer                 :: i
```

3.2 Bracketing Root by Creating Subintervals Within Larger Interval

`inbracket(func, intvl, n_seg, sub_intvl, n_root)`

3.2.1 Arguments

```
subroutine inbracket(func, intvl, n_seg, sub_intvl, n_root)
  !=======================================================!
  ! Find intervals which enclose a given # of roots       !
  !-------------------------------------------------------!
  ! Cycles through a number of iterations until the       !
  ! number of proposed roots is reached. A master interval!
  ! is traversed, and subintervals are generated when a   !
  ! root is found (function changes sign). Outputs the    !
  ! intervals found and the number of roots found.        !
  !-------------------------------------------------------!
  ! Inputs:                                               !
  ! func  = function whose roots we're trying to bracket  !
  ! intvl = master interval (contains the subints)        !
  ! n_seg = number of segments to analyse                 !
  !-------------------------------------------------------!
  ! Outputs:                                              !
  ! sub_intvl = subintervals                              !
  !-------------------------------------------------------!
  ! Inputs-Outputs:                                       !
  ! n_root() = # of roots found                           !
  !-------------------------------------------------------!
  ! Locals:                                               !
  ! f()     = value of func() at the subinterval bounds   !
  ! x       = independent variable                        !
  ! dx      = step size                                   !
  ! i       = counter                                     !
  ! n_intvl = # of intervals                              !
```

```
!=======================================================!
implicit none
interface cont_func
  function func(x)
    use nrtype
    implicit none
    real(dp), intent(in) :: x
    real(dp)             :: func
  end function func
end interface cont_func
real(dp), intent(in)    :: intvl(2)
integer,  intent(in)    :: n_seg
real(dp), intent(out)   :: sub_intvl(:,:)
integer,  intent(inout) :: n_root
real(dp)                :: f(2), x, dx
integer                 :: i, n_intvl
```