fieldfun.m Examples

Example 1a

Concatenation of fields in a structure array and scalar structure containing mixed data types.

```
treatmentGroup(1) = struct( 'name', "John", 'age', 30, ...
    'vaccinated', true );
treatmentGroup(2) = struct( 'name', "Jane", 'age', 80, ...
    'vaccinated', false );
controlGroup = struct( 'name', "Jim", 'age', 50, 'vaccinated', true );
allParticipants = fieldfun( @horzcat, treatmentGroup, controlGroup )
```

```
allParticipants = struct with fields:
          name: ["John"    "Jane"    "Jim"]
           age: [30 80 50]
    vaccinated: [1 0 1]
```

```
averageAge = mean(combinedPatients.age)
```

```
averageAge = 53.3333
```

Example 1b

Custom function, demonstrating the use of *varargin* to accept fields from a variable number of structures. For each function call, *varargin* provides the values for all structures of a given field as a cell array.

```
fun = @(varargin) replace( string( varargin ), ...
    alphanumericsPattern(1), "#" );
redactedParticipants = fieldfun( fun, treatmentGroup, controlGroup )
```

```
redactedParticipants = struct with fields:
          name: ["####"    "####"    "###"]
           age: ["##"    "##"    "##"]
    vaccinated: ["####"    "#####"    "####"]
```

Example 2

Custom function, demonstrating how to pass multiple field values to a function which accepts data as an array.

```
IsWorkDone = struct( 'methods', true, 'results', true );
IsReportWritten = struct( 'methods', true, 'results', true );
IsChecked = struct( 'methods', true, 'results', false );
fun = @(varargin) all([varargin{:}]);
IsComplete = fieldfun( fun, IsWorkDone, IsReportWritten, IsChecked )
```

```
IsComplete = struct with fields:
    methods: 1
    results: 0
```

Example 3

Demonstrates logical (boolean) operations on the fields of a structure. Demonstrates use of strings to define function name via MATLAB operators, chaining *fieldfun* calls, and scalar inputs.

```
IsRaining = struct( 'Sun', false, 'Mon', false, 'Tue', true );
isHoliday = true;
IsNotRaining = fieldfun( "~", IsRaining );
IsGoToPark = fieldfun( "&", IsNotRaining, isHoliday )
```

```
IsGoToPark = struct with fields:
    Sun: 1
    Mon: 1
    Tue: 0
```

Note that the *fieldfun* call to calculate *IsNotRaining* takes only one structure. Compared to *structfun*, the output is also a structure rather than an array. Note that *isHoliday* is not a structure but a scalar logical which is implictly converted to fill all the equivalent fields of *IsNotRaining*.