

# Using the reshape function

The R Core Team

May 13, 2024

## 1 Introduction

The `reshape()` function reshapes datasets in the so-called ‘wide’ format (with repeated measurements in separate columns of the same row) to the ‘long’ format (with the repeated measurements in separate rows), and vice versa.

`reshape()` is a somewhat complicated function, and this vignette gives a few examples of how it can be used. Although `reshape()` can be used in a variety of contexts, the motivating application is data from longitudinal studies, and the arguments of this function are named and described in those terms. See the documentation (`help(reshape)`) for background and detailed usage.

For our examples, we will simulate data from a study where individuals are measured at two time points. Two of the measurements are time-varying: height and weight, and one of the measurements is time-constant: sex.

## 2 Conversion from wide to long format

We first simulate data in the wide format. Data from each individual is contained in one row, with one column for time-constant variables and multiple columns for time-varying variables. Here there are two time points (before and after), so there are two columns for each time-varying variable.

```
> set.seed(12345)
> n <- 5
> d1 <- data.frame(sex = sample(c("M", "F"), n, rep = TRUE),
                   ht.before = round(rnorm(n, 165, 6), 1),
                   ht.after = round(rnorm(n, 165, 6), 1),
                   wt.before = round(rnorm(n, 80, 6)),
                   wt.after = round(rnorm(n, 80, 6)))
> d1
```

|   | sex | ht.before | ht.after | wt.before | wt.after |
|---|-----|-----------|----------|-----------|----------|
| 1 | F   | 159.2     | 164.4    | 78        | 78       |
| 2 | M   | 165.1     | 163.5    | 81        | 88       |
| 3 | F   | 178.9     | 175.0    | 80        | 73       |
| 4 | F   | 158.8     | 162.3    | 65        | 79       |
| 5 | F   | 146.7     | 168.3    | 83        | 84       |

Suppose we want to convert this dataset into the long format, with two rows for each individual, and one column for each variable (both time-constant and time-varying). Such a representation will need two additional variables to distinguish between multiple rows corresponding to the same individual (corresponding to one row in the wide format): a time-variable and an id-variable. These will be automatically created when converting from wide to long format.

However, we do need to specify which columns in the wide format correspond to the same time-varying variable(s). This is easiest to do when we have only one time-varying variable. Although we have two such in our example, let us pretend that only height is time-varying. The corresponding columns can be specified as the `varying` argument. The two weight variables will then be assumed to be different time-constant variables, similar to sex.

```
> reshape(d1, direction = "long",
           varying = c("ht.before", "ht.after"))
```

|          | sex | wt.before | wt.after | time   | ht    | id |
|----------|-----|-----------|----------|--------|-------|----|
| 1.before | F   | 78        | 78       | before | 159.2 | 1  |
| 2.before | M   | 81        | 88       | before | 165.1 | 2  |
| 3.before | F   | 80        | 73       | before | 178.9 | 3  |
| 4.before | F   | 65        | 79       | before | 158.8 | 4  |
| 5.before | F   | 83        | 84       | before | 146.7 | 5  |
| 1.after  | F   | 78        | 78       | after  | 164.4 | 1  |
| 2.after  | M   | 81        | 88       | after  | 163.5 | 2  |
| 3.after  | F   | 80        | 73       | after  | 175.0 | 3  |
| 4.after  | F   | 65        | 79       | after  | 162.3 | 4  |
| 5.after  | F   | 83        | 84       | after  | 168.3 | 5  |

It is equivalent to specify the variables as column indices.

```
> reshape(d1, direction = "long",
           varying = c(2, 3))
```

|          | sex | wt.before | wt.after | time   | ht    | id |
|----------|-----|-----------|----------|--------|-------|----|
| 1.before | F   | 78        | 78       | before | 159.2 | 1  |
| 2.before | M   | 81        | 88       | before | 165.1 | 2  |
| 3.before | F   | 80        | 73       | before | 178.9 | 3  |
| 4.before | F   | 65        | 79       | before | 158.8 | 4  |
| 5.before | F   | 83        | 84       | before | 146.7 | 5  |
| 1.after  | F   | 78        | 78       | after  | 164.4 | 1  |
| 2.after  | M   | 81        | 88       | after  | 163.5 | 2  |
| 3.after  | F   | 80        | 73       | after  | 175.0 | 3  |
| 4.after  | F   | 65        | 79       | after  | 162.3 | 4  |
| 5.after  | F   | 83        | 84       | after  | 168.3 | 5  |

Note that the names of the combined variable, as well as the values of the time variable, are automatically detected because the names happen to be “nicely” formatted. Suppose we instead had

```
> n <- 5
> d2 <- data.frame(sex = sample(c("M", "F"), n, rep = TRUE),
```

```

ht_before = round(rnorm(n, 165, 6), 1),
ht_after = round(rnorm(n, 165, 6), 1),
wt_before = round(rnorm(n, 80, 6)),
wt_after = round(rnorm(n, 80, 6))

```

Modifying the previous call gives:

```

> try(
  reshape(d2, direction = "long",
    varying = c("wt_before", "wt_after")),
)

```

```

Error in guess(varying) :
  failed to guess time-varying variables from their names

```

This is easy to “fix” in this case because the names are still nicely formatted, just not using the separator that `reshape()` expects by default.

```

> reshape(d2, direction = "long",
  varying = c("wt_before", "wt_after"), sep = "_")

```

|          | sex | ht_before | ht_after | time   | wt | id |
|----------|-----|-----------|----------|--------|----|----|
| 1.before | F   | 175.8     | 169.9    | before | 83 | 1  |
| 2.before | M   | 162.1     | 178.2    | before | 78 | 2  |
| 3.before | F   | 168.7     | 177.3    | before | 70 | 3  |
| 4.before | M   | 168.7     | 174.8    | before | 91 | 4  |
| 5.before | F   | 164.0     | 166.5    | before | 80 | 5  |
| 1.after  | F   | 175.8     | 169.9    | after  | 87 | 1  |
| 2.after  | M   | 162.1     | 178.2    | after  | 66 | 2  |
| 3.after  | F   | 168.7     | 177.3    | after  | 74 | 3  |
| 4.after  | M   | 168.7     | 174.8    | after  | 86 | 4  |
| 5.after  | F   | 164.0     | 166.5    | after  | 85 | 5  |

A more general solution is to specify the name of the new combined column explicitly as the `v.names` argument.

```

> reshape(d2, direction = "long",
  varying = c("wt_before", "wt_after"),
  v.names = "weight")

```

|     | sex | ht_before | ht_after | time | weight | id |
|-----|-----|-----------|----------|------|--------|----|
| 1.1 | F   | 175.8     | 169.9    | 1    | 83     | 1  |
| 2.1 | M   | 162.1     | 178.2    | 1    | 78     | 2  |
| 3.1 | F   | 168.7     | 177.3    | 1    | 70     | 3  |
| 4.1 | M   | 168.7     | 174.8    | 1    | 91     | 4  |
| 5.1 | F   | 164.0     | 166.5    | 1    | 80     | 5  |
| 1.2 | F   | 175.8     | 169.9    | 2    | 87     | 1  |
| 2.2 | M   | 162.1     | 178.2    | 2    | 66     | 2  |
| 3.2 | F   | 168.7     | 177.3    | 2    | 74     | 3  |
| 4.2 | M   | 168.7     | 174.8    | 2    | 86     | 4  |
| 5.2 | F   | 164.0     | 166.5    | 2    | 85     | 5  |

We can additionally specify the names and values of the id / time variables as well.

```
> reshape(d2, direction = "long",
          varying = c("wt_before", "wt_after"),
          v.names = "weight",
          timevar = "when", times = c("pre", "post"),
          idvar = "subject", ids = letters[1:n])
```

|        | sex | ht_before | ht_after | when | weight | subject |
|--------|-----|-----------|----------|------|--------|---------|
| a.pre  | F   | 175.8     | 169.9    | pre  | 83     | a       |
| b.pre  | M   | 162.1     | 178.2    | pre  | 78     | b       |
| c.pre  | F   | 168.7     | 177.3    | pre  | 70     | c       |
| d.pre  | M   | 168.7     | 174.8    | pre  | 91     | d       |
| e.pre  | F   | 164.0     | 166.5    | pre  | 80     | e       |
| a.post | F   | 175.8     | 169.9    | post | 87     | a       |
| b.post | M   | 162.1     | 178.2    | post | 66     | b       |
| c.post | F   | 168.7     | 177.3    | post | 74     | c       |
| d.post | M   | 168.7     | 174.8    | post | 86     | d       |
| e.post | F   | 164.0     | 166.5    | post | 85     | e       |

Note that the `times` argument is ignored when automatic guessing is performed, i.e., when `v.names` is not explicitly specified.

```
> reshape(d2, direction = "long",
          varying = c("wt_before", "wt_after"), sep = "_",
          ## v.names = "wt", # without this, 'times' is unused
          timevar = "when", times = c("pre", "post"))
```

|          | sex | ht_before | ht_after | when   | wt | id |
|----------|-----|-----------|----------|--------|----|----|
| 1.before | F   | 175.8     | 169.9    | before | 83 | 1  |
| 2.before | M   | 162.1     | 178.2    | before | 78 | 2  |
| 3.before | F   | 168.7     | 177.3    | before | 70 | 3  |
| 4.before | M   | 168.7     | 174.8    | before | 91 | 4  |
| 5.before | F   | 164.0     | 166.5    | before | 80 | 5  |
| 1.after  | F   | 175.8     | 169.9    | after  | 87 | 1  |
| 2.after  | M   | 162.1     | 178.2    | after  | 66 | 2  |
| 3.after  | F   | 168.7     | 177.3    | after  | 74 | 3  |
| 4.after  | M   | 168.7     | 174.8    | after  | 86 | 4  |
| 5.after  | F   | 164.0     | 166.5    | after  | 85 | 5  |

So far, we have only specified one time-varying variable, but our data actually has two. How do we specify multiple time-varying variables? This depends on whether the variable names are in a guessable format.

## 2.1 Explicitly specifying variables names

The general approach is to explicitly specify both `varying` and `v.names` as before. `v.names` should be a vector of new variable names in the long format, and `varying` should either be a list, with each component giving the corresponding wide format variable names, or a matrix, with each row giving the corresponding wide format variable names.

```
> reshape(d2, direction = "long",
          varying = list(c("ht_before", "ht_after"),
                        c("wt_before", "wt_after")), # list form
          v.names = c("height", "weight"),
          times = c("pre", "post"))
```

|        | sex | time | height | weight | id |
|--------|-----|------|--------|--------|----|
| 1.pre  | F   | pre  | 175.8  | 83     | 1  |
| 2.pre  | M   | pre  | 162.1  | 78     | 2  |
| 3.pre  | F   | pre  | 168.7  | 70     | 3  |
| 4.pre  | M   | pre  | 168.7  | 91     | 4  |
| 5.pre  | F   | pre  | 164.0  | 80     | 5  |
| 1.post | F   | post | 169.9  | 87     | 1  |
| 2.post | M   | post | 178.2  | 66     | 2  |
| 3.post | F   | post | 177.3  | 74     | 3  |
| 4.post | M   | post | 174.8  | 86     | 4  |
| 5.post | F   | post | 166.5  | 85     | 5  |

```
> reshape(d2, direction = "long",
          varying = rbind(c("ht_before", "ht_after"),
                        c("wt_before", "wt_after")), # matrix form
          v.names = c("height", "weight"))
```

|     | sex | time | height | weight | id |
|-----|-----|------|--------|--------|----|
| 1.1 | F   | 1    | 175.8  | 83     | 1  |
| 2.1 | M   | 1    | 162.1  | 78     | 2  |
| 3.1 | F   | 1    | 168.7  | 70     | 3  |
| 4.1 | M   | 1    | 168.7  | 91     | 4  |
| 5.1 | F   | 1    | 164.0  | 80     | 5  |
| 1.2 | F   | 2    | 169.9  | 87     | 1  |
| 2.2 | M   | 2    | 178.2  | 66     | 2  |
| 3.2 | F   | 2    | 177.3  | 74     | 3  |
| 4.2 | M   | 2    | 174.8  | 86     | 4  |
| 5.2 | F   | 2    | 166.5  | 85     | 5  |

The `times` argument has been omitted in the second example above, and the default is to use sequential times. The `v.names` argument can be omitted as well, but the default is not generally sensible.

Of course, the time and id variables can also be controlled in the usual way as long as `v.names` is specified.

```
> reshape(d2, direction = "long",
          varying = rbind(c("ht_before", "ht_after"),
                        c("wt_before", "wt_after")),
          v.names = c("height", "weight"),
          timevar = "when",
          times = c("pre", "post"),
          idvar = "subject",
          ids = letters[1:n])
```

|        | sex | when | height | weight | subject |
|--------|-----|------|--------|--------|---------|
| a.pre  | F   | pre  | 175.8  | 83     | a       |
| b.pre  | M   | pre  | 162.1  | 78     | b       |
| c.pre  | F   | pre  | 168.7  | 70     | c       |
| d.pre  | M   | pre  | 168.7  | 91     | d       |
| e.pre  | F   | pre  | 164.0  | 80     | e       |
| a.post | F   | post | 169.9  | 87     | a       |
| b.post | M   | post | 178.2  | 66     | b       |
| c.post | F   | post | 177.3  | 74     | c       |
| d.post | M   | post | 174.8  | 86     | d       |
| e.post | F   | post | 166.5  | 85     | e       |

## 2.2 Variables names in a guessable format

Even when variable names are in a guessable format, `reshape()` will not try to guess if multiple time-varying variables are provided as a list or matrix. However, when the wide format variable names are suitably formatted in the same manner for all time-varying variables, it is still possible to take advantage of automatic guessing by specifying the `varying` argument as an atomic vector (of either names or indices) containing all time-varying columns.

```
> reshape(d2, direction = "long",
           varying = c("ht_before", "ht_after",
                       "wt_before", "wt_after"), sep = "_")
```

|          | sex | time   | ht    | wt | id |
|----------|-----|--------|-------|----|----|
| 1.before | F   | before | 175.8 | 83 | 1  |
| 2.before | M   | before | 162.1 | 78 | 2  |
| 3.before | F   | before | 168.7 | 70 | 3  |
| 4.before | M   | before | 168.7 | 91 | 4  |
| 5.before | F   | before | 164.0 | 80 | 5  |
| 1.after  | F   | after  | 169.9 | 87 | 1  |
| 2.after  | M   | after  | 178.2 | 66 | 2  |
| 3.after  | F   | after  | 177.3 | 74 | 3  |
| 4.after  | M   | after  | 174.8 | 86 | 4  |
| 5.after  | F   | after  | 166.5 | 85 | 5  |

The atomic vector form of `varying` can be combined with explicit (non-guessed) specification of `v.names` as well, but in that case, one needs to pay careful attention to the order of variable names in `varying`. The following gives wrong results:

```
> reshape(d2, direction = "long",
           varying = c("ht_before", "ht_after",
                       "wt_before", "wt_after"),
           v.names = c("height", "weight"))
```

|     | sex | time | height | weight | id |
|-----|-----|------|--------|--------|----|
| 1.1 | F   | 1    | 175.8  | 169.9  | 1  |
| 2.1 | M   | 1    | 162.1  | 178.2  | 2  |
| 3.1 | F   | 1    | 168.7  | 177.3  | 3  |
| 4.1 | M   | 1    | 168.7  | 174.8  | 4  |

|     |   |   |       |       |   |
|-----|---|---|-------|-------|---|
| 5.1 | F | 1 | 164.0 | 166.5 | 5 |
| 1.2 | F | 2 | 83.0  | 87.0  | 1 |
| 2.2 | M | 2 | 78.0  | 66.0  | 2 |
| 3.2 | F | 2 | 70.0  | 74.0  | 3 |
| 4.2 | M | 2 | 91.0  | 86.0  | 4 |
| 5.2 | F | 2 | 80.0  | 85.0  | 5 |

The correct order requires all columns corresponding to the same time to be contiguous; this is the same intrinsic column-major ordering in the matrix form above. It is best to avoid the atomic vector form of `varying` unless `v.names` is being omitted.

## 2.3 Repeated application of reshape

Just as an illustration, let us try to create an even longer dataset that combines height and weight together in a single column.

```
> dlong <-
  reshape(d2, direction = "long",
    varying = c("ht_before", "wt_before",
      "ht_after", "wt_after"),
    v.names = c("height", "weight"),
    timevar = "when", times = c("pre", "post"),
    idvar = "subject", ids = letters[1:n])
> reshape(dlong, direction = "long",
  varying = c("height", "weight"),
  v.names = "combined",
  timevar = "what", times = c("height", "weight"))
```

|           | sex | when | subject | what   | combined | id |
|-----------|-----|------|---------|--------|----------|----|
| 1.height  | F   | pre  | a       | height | 175.8    | 1  |
| 2.height  | M   | pre  | b       | height | 162.1    | 2  |
| 3.height  | F   | pre  | c       | height | 168.7    | 3  |
| 4.height  | M   | pre  | d       | height | 168.7    | 4  |
| 5.height  | F   | pre  | e       | height | 164.0    | 5  |
| 6.height  | F   | post | a       | height | 169.9    | 6  |
| 7.height  | M   | post | b       | height | 178.2    | 7  |
| 8.height  | F   | post | c       | height | 177.3    | 8  |
| 9.height  | M   | post | d       | height | 174.8    | 9  |
| 10.height | F   | post | e       | height | 166.5    | 10 |
| 1.weight  | F   | pre  | a       | weight | 83.0     | 1  |
| 2.weight  | M   | pre  | b       | weight | 78.0     | 2  |
| 3.weight  | F   | pre  | c       | weight | 70.0     | 3  |
| 4.weight  | M   | pre  | d       | weight | 91.0     | 4  |
| 5.weight  | F   | pre  | e       | weight | 80.0     | 5  |
| 6.weight  | F   | post | a       | weight | 87.0     | 6  |
| 7.weight  | M   | post | b       | weight | 66.0     | 7  |
| 8.weight  | F   | post | c       | weight | 74.0     | 8  |
| 9.weight  | M   | post | d       | weight | 86.0     | 9  |
| 10.weight | F   | post | e       | weight | 85.0     | 10 |

Can we get this directly from `d2` using a single `reshape()` call? We can, except that we will get a composite time variable (which can be easily split if needed).

```
> reshape(d2, direction = "long",
          v.names = "combined",
          varying = c("ht_before", "ht_after", "wt_before", "wt_after"),
          timevar = "when_what",
          times = c("pre_height", "post_height", "pre_weight", "post_weight"),
          idvar = "subject", ids = letters[1:n])
```

|               | sex | when_what   | combined | subject |
|---------------|-----|-------------|----------|---------|
| a.pre_height  | F   | pre_height  | 175.8    | a       |
| b.pre_height  | M   | pre_height  | 162.1    | b       |
| c.pre_height  | F   | pre_height  | 168.7    | c       |
| d.pre_height  | M   | pre_height  | 168.7    | d       |
| e.pre_height  | F   | pre_height  | 164.0    | e       |
| a.post_height | F   | post_height | 169.9    | a       |
| b.post_height | M   | post_height | 178.2    | b       |
| c.post_height | F   | post_height | 177.3    | c       |
| d.post_height | M   | post_height | 174.8    | d       |
| e.post_height | F   | post_height | 166.5    | e       |
| a.pre_weight  | F   | pre_weight  | 83.0     | a       |
| b.pre_weight  | M   | pre_weight  | 78.0     | b       |
| c.pre_weight  | F   | pre_weight  | 70.0     | c       |
| d.pre_weight  | M   | pre_weight  | 91.0     | d       |
| e.pre_weight  | F   | pre_weight  | 80.0     | e       |
| a.post_weight | F   | post_weight | 87.0     | a       |
| b.post_weight | M   | post_weight | 66.0     | b       |
| c.post_weight | F   | post_weight | 74.0     | c       |
| d.post_weight | M   | post_weight | 86.0     | d       |
| e.post_weight | F   | post_weight | 85.0     | e       |

### 3 Conversion from long to wide format

Conversion from long to wide format is generally simpler. Let us simulate long format data from the same hypothetical setup.

```
> d3 <- data.frame(sex = sample(c("M", "F"), 2 * n, rep = TRUE),
                  ht = round(rnorm(2 * n, 165, 6), 1),
                  wt = round(rnorm(2 * n, 80, 6)),
                  subject = rep(1:n, 2),
                  when = rep(c("pre", "post"), each = n))
> d3
```

|   | sex | ht    | wt | subject | when |
|---|-----|-------|----|---------|------|
| 1 | M   | 161.8 | 81 | 1       | pre  |
| 2 | M   | 176.7 | 72 | 2       | pre  |
| 3 | F   | 165.3 | 83 | 3       | pre  |
| 4 | M   | 167.1 | 90 | 4       | pre  |



|    |   |       |    |   |      |
|----|---|-------|----|---|------|
| 5  | F | 161.0 | 76 | 5 | pre  |
| 6  | F | 166.7 | 69 | 1 | post |
| 7  | F | 169.1 | 85 | 2 | post |
| 8  | M | 169.9 | 90 | 3 | post |
| 9  | F | 177.9 | 83 | 4 | post |
| 10 | F | 150.9 | 72 | 5 | post |

To convert this to the wide format, the arguments `idvar` and `timevar` to `reshape()` are mandatory, and all other variables are assumed to be time-varying. This is what we do in the next example, where even `sex` is erroneously treated as time-varying.

```
> reshape(d3, direction = "wide",
           idvar = "subject", timevar = "when")
```

|   | subject | sex.pre | ht.pre | wt.pre | sex.post | ht.post | wt.post |
|---|---------|---------|--------|--------|----------|---------|---------|
| 1 | 1       | M       | 161.8  | 81     | F        | 166.7   | 69      |
| 2 | 2       | M       | 176.7  | 72     | F        | 169.1   | 85      |
| 3 | 3       | F       | 165.3  | 83     | M        | 169.9   | 90      |
| 4 | 4       | M       | 167.1  | 90     | F        | 177.9   | 83      |
| 5 | 5       | F       | 161.0  | 76     | F        | 150.9   | 72      |

To specify some variables as time-constant, the time-varying variables must be explicitly specified through `v.names`.

```
> reshape(d3, direction = "wide",
           idvar = "subject", timevar = "when",
           v.names = c("ht", "wt"))
```

This gives a warning because `sex` is not really time-constant in the dataset we have created. Let us fix that:

```
> n <- 10
> d4 <- data.frame(sex = rep(sample(c("M", "F"), n, rep = TRUE), 2),
                  ht = round(rnorm(2 * n, 165, 6), 1),
                  wt = round(rnorm(2 * n, 80, 6)),
                  subject = rep(1:n, 2),
                  when = rep(c("pre", "post"), each = n))
> reshape(d4, direction = "wide",
           idvar = "subject", timevar = "when",
           v.names = c("ht", "wt"), sep = "_")
```

|    | sex | subject | ht_pre | wt_pre | ht_post | wt_post |
|----|-----|---------|--------|--------|---------|---------|
| 1  | F   | 1       | 170.7  | 84     | 176.2   | 77      |
| 2  | F   | 2       | 170.0  | 77     | 169.0   | 78      |
| 3  | F   | 3       | 160.1  | 93     | 163.2   | 89      |
| 4  | F   | 4       | 167.9  | 76     | 168.2   | 77      |
| 5  | M   | 5       | 171.1  | 76     | 169.9   | 82      |
| 6  | F   | 6       | 168.9  | 81     | 159.2   | 73      |
| 7  | F   | 7       | 171.3  | 73     | 159.9   | 72      |
| 8  | M   | 8       | 163.2  | 83     | 176.3   | 88      |
| 9  | M   | 9       | 179.9  | 72     | 162.6   | 88      |
| 10 | F   | 10      | 170.8  | 81     | 159.1   | 80      |

To specify the resulting wide format variable names explicitly instead of using the automatically constructed defaults, we may use the `varying` argument as in wide-to-long conversion. As in that case, `varying` can be a vector of variable names, where the same caveats apply regarding order.

```
> reshape(d4, direction = "wide",
           idvar = "subject", timevar = "when",
           v.names = c("ht", "wt"),
           varying = c("h_before", "w_before", "h_after", "w_after"))
```

|    | sex | subject | h_before | w_before | h_after | w_after |
|----|-----|---------|----------|----------|---------|---------|
| 1  | F   | 1       | 170.7    | 84       | 176.2   | 77      |
| 2  | F   | 2       | 170.0    | 77       | 169.0   | 78      |
| 3  | F   | 3       | 160.1    | 93       | 163.2   | 89      |
| 4  | F   | 4       | 167.9    | 76       | 168.2   | 77      |
| 5  | M   | 5       | 171.1    | 76       | 169.9   | 82      |
| 6  | F   | 6       | 168.9    | 81       | 159.2   | 73      |
| 7  | F   | 7       | 171.3    | 73       | 159.9   | 72      |
| 8  | M   | 8       | 163.2    | 83       | 176.3   | 88      |
| 9  | M   | 9       | 179.9    | 72       | 162.6   | 88      |
| 10 | F   | 10      | 170.8    | 81       | 159.1   | 80      |

For more than one time-varying variable, it is safer to avoid the vector form and instead specify `varying` as a list or matrix.

```
> reshape(d4, direction = "wide",
           idvar = "subject", timevar = "when",
           v.names = c("ht", "wt"),
           varying = list(c("h_before", "h_after"),
                          c("w_before", "w_after")))
```

|    | sex | subject | h_before | w_before | h_after | w_after |
|----|-----|---------|----------|----------|---------|---------|
| 1  | F   | 1       | 170.7    | 84       | 176.2   | 77      |
| 2  | F   | 2       | 170.0    | 77       | 169.0   | 78      |
| 3  | F   | 3       | 160.1    | 93       | 163.2   | 89      |
| 4  | F   | 4       | 167.9    | 76       | 168.2   | 77      |
| 5  | M   | 5       | 171.1    | 76       | 169.9   | 82      |
| 6  | F   | 6       | 168.9    | 81       | 159.2   | 73      |
| 7  | F   | 7       | 171.3    | 73       | 159.9   | 72      |
| 8  | M   | 8       | 163.2    | 83       | 176.3   | 88      |
| 9  | M   | 9       | 179.9    | 72       | 162.6   | 88      |
| 10 | F   | 10      | 170.8    | 81       | 159.1   | 80      |