In this example we will learn how to use `fitPS` to fit a zeta distribution to some data from a survey where the number of groups of glass found is recorded. The data in this example comes from @roux2001 who surveyed the footwear of 776 individuals in south-eastern Australia, and is summarised in the table below.

n	r_n
0	754
1	9
2	8
3	4
4	1

This data set is built into the package and can be accessed from the `Psurveys` object. That is, we can type:

```
> data("Psurveys")
> roux = Psurveys$roux
```

The data is stored as an object of class `psData`. This probably will not be of importance to most users. If you are interested in the details, then these can be found in the **Value** section of the help page for `readData`. There is an S3 `print` method for objects of time, meaning that if we print the object—either by typing its name at the command prompt, or by explicitly calling `print`—then we will get formatted printing of the information contained within the object. Specifically:

- the values of n and r_n will be printed out in tabular format (where n is the number of groups or the size of the groups, and r_n is the number of times n has been observed in the survey),
- the type of survey will be printed—either "Number of Groups" or "Group Size",
- and if the object has a reference or notes attached to it, then these will be printed as well

For example

```
> roux

Number of Groups

  n    rn
---  ----
  0   754
  1     9
```

```
  2     8
  3     4
  4     1
Roux C, Kirk R, Benson S, Van Haren T, Petterd C (2001).
"Glass particles in footwear of members of the public in
south-eastern Australia-a survey." _Forensic Science
International_, *116*(2), 149-156.
doi:10.1016/S0379-0738(00)00355-8
<https://doi.org/10.1016/S0379-0738%2800%2900355-8>.
```

It is very simple to fit a zeta distribution to this data set. We do this using the `fitDist` function.

```
> fit = fitDist(roux)
```

The function returns an object of class `psFit`, the details of which can be found in the help page for `fitDist`. There are both S3 `print` and S3 `plot` methods for objects of this class. The `print` method method displays an estimate of the shape parameter s, an estimate of the standard deviation—the standard error—of the estimate of s ($\widehat{\text{sd}}(\hat{s}) = \text{se}(\hat{s})$). **NOTE**: it is important to understand that the value of the shape parameter that is displayed, and the value that is stored in the fitted object differ by 1. That is, s is shown,m and $s' = s - 1$ is stored. This is done because the package which assists in the fitting, `VGAM`, is parameterised in terms of s'. This difference only has consequences if the fitted value is being used in conjuction with other `VGAM` functions. The `print` method also displays the first 10 fitted probabilities from the model by default.

```
> fit

The estimated shape parameter is 4.9544
The standard error of shape parameter is 0.2366
------
NOTE: The shape parameter is reported so that it is consistent
with Coulson et al. However, the value returned is actually s'
= shape - 1 to be consistent with the VGAM parameterisation,
which is used for computation. This has flow on effects, for
example in confInt. This will be changed at some point.
------

The first  10 fitted values are:
          P0           P1           P2           P3           P4
9.631547e-01 3.106447e-02 4.167082e-03 1.001917e-03 3.316637e-04
          P5           P6           P7           P8           P9
1.344002e-04 6.262053e-05 3.231467e-05 1.802885e-05 1.069709e-05
```

The package provides a `confint` method for the fitted value. The method returns both a Wald confidence interval and profile likelihood interval. The Wald

interval takes the usual the form where the lower and upper bound are given by $\hat{s} \pm z^*(1-\alpha/2) \times se(\hat{s})$). The profile likelihood interval finds the end-points of the interval that satisfies

$$-2\left[\ell(\hat{s};\mathbf{x}) - \ell(s;\mathbf{x})\right] \leq \chi^2_1(\alpha)$$

where $\ell(s;\mathbf{x})$ is value of the log-likelihood given shape parameter s. The two intervals are returned as elements of a `list` named `wald` and `prof` respectively.

```
> ci = confint(fit)
> ci$wald

    2.5%    97.5%
3.490761 4.418099

> ci$prof

    2.5%    97.5%
3.520495 4.451277
```

You will notice that neither of these intervals contain the value shown in the previous output. However, this simply because they are confidence intervals on s' and not s. This can be remedied by adding one to each interval:

```
> ci$wald + 1

    2.5%    97.5%
4.490761 5.418099

> ci$prof + 1

    2.5%    97.5%
4.520495 5.451277
```

The reason for not *correcting* these intervals is that the method mostly exists to feed into other parts of the package, especially the `plot` method.