

# Package ‘lcqc’

January 27, 2025

**Title** Chromatographic Peak Detection, Analysis, and Reporting

**Version** 0.9.7

**Description** LCQC is used to automatically detect, process, and analyze chromatographic peaks for various metrics relevant to quality control of liquid chromatographic (HPLC) columns. Functions for baseline correction, chromatogram smoothing, peak detection and integration, non-linear iterative curve fitting (peak deconvolution), performance metric calculations, and automated reporting are included.

**License** MIT + file LICENSE

**URL** <https://github.com/Deniz-Koseoglu/lcqc>

**Depends** R (>= 4.0.0)

**Imports** utils,  
stats,  
graphics,  
grDevices,  
stringr,  
fs (>= 1.6.3),  
devtools (>= 2.4.5),  
data.table (>= 1.14.8),  
ggplot2 (>= 3.4.4),  
ggpubr (>= 0.6.0),  
scales (>= 1.2.1),  
nloptr (>= 2.0.3),  
akima (>= 0.6.3.4),  
quarto (>= 1.4),  
flextable (>= 0.9.5),  
tibble (>= 3.2.1),  
knitr (>= 1.45),  
rlang,  
lifecycle,  
glue

**Suggests** testthat (>= 3.2.0),  
rmarkdown (>= 2.25)

**SystemRequirements** Quarto (<https://quarto.org/docs/download/>)

**VignetteBuilder** knitr

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.2

**Config/testthat/edition** 3

**LazyData** true

**BugReports** <https://github.com/Deniz-Koseoglu/lcqc/issues>

## Contents

acc_inf . . . . .	3
acc_max . . . . .	4
addnms . . . . .	5
blinc_als . . . . .	6
blinc_chang . . . . .	7
blinc_isrea . . . . .	8
blinc_plot . . . . .	9
blinc_poly . . . . .	10
browse_visc . . . . .	11
chk_pack . . . . .	13
chrom_addmets . . . . .	14
chrom_amplim . . . . .	15
chrom_asym . . . . .	17
chrom_bline . . . . .	19
chrom_deriv . . . . .	20
chrom_derlims . . . . .	21
chrom_detect . . . . .	23
chrom_export . . . . .	26
chrom_icf . . . . .	28
chrom_plot . . . . .	31
chrom_res . . . . .	33
chrom_retfr . . . . .	35
chrom_sepf . . . . .	36
chrom_skim . . . . .	37
chrom_smooth . . . . .	41
chrom_tpa . . . . .	42
chrom_tplate . . . . .	44
chrom_width . . . . .	47
cprint . . . . .	49
dtprer . . . . .	49
exgc1 . . . . .	50
exgc2 . . . . .	51
exlc1 . . . . .	51
exlc10 . . . . .	52
exlc11 . . . . .	52
exlc2 . . . . .	53
exlc3 . . . . .	53
exlc4 . . . . .	54
exlc5 . . . . .	54
exlc6 . . . . .	55
exlc7 . . . . .	55
exlc8 . . . . .	56
exlc9 . . . . .	56

*acc\_inf* 3

ext_lcqc . . . . .	57
fastchrom_bline . . . . .	57
find_peaks . . . . .	58
icf_plot . . . . .	59
insect . . . . .	61
integ_plot . . . . .	61
is.even . . . . .	62
linspline . . . . .	63
noise_plot . . . . .	64
read_shim . . . . .	65
render_defs . . . . .	67
sdmin_infchk . . . . .	69
simlc1 . . . . .	70
simlc2 . . . . .	70
simlc3 . . . . .	71
simlc4 . . . . .	71
simlc5 . . . . .	72
simlc6 . . . . .	72
simlc7 . . . . .	73
slope . . . . .	73
smooth_whit . . . . .	74
supp_pars . . . . .	74
z_optim . . . . .	75
z_thres . . . . .	76

**Index** 78

---

<i>acc_inf</i>	<i>Interpolate accurate inflection points</i>
----------------	---

---

## Description

Interpolates the exact xy-coordinates (i.e. retention time and signal) of inflection point from [chrom\\_detect](#) approximation by surveying a neighbourhood of nearby points.

## Usage

```
acc_inf(inds = seq_along(xvals), xvals, yvals, sd, linfs, rinfs)
```

## Arguments

<code>inds</code>	Optional numeric vector of data indices (by default, the sequence along retention time data is used, i.e. <code>seq_along(xvals)</code> ).
<code>xvals</code>	Retention time (x-axis) data as a numeric vector.
<code>yvals</code>	Signal (y-axis) data as a numeric vector.
<code>sd</code>	Second derivative of <code>yvals</code> as a numeric vector.
<code>linfs, rinfs</code>	Indices of <b>left</b> ( <code>linfs</code> ) and <b>right</b> ( <code>rinfs</code> ) inflection points (e.g. approximated by <a href="#">chrom_detect</a> ).

**Value**

A list of two data.frame objects containing left and right inflection point coordinates ("acc\_x" and acc\_y) as well as the data indices between which these are situated ("low" and "high").

**See Also**

[acc\\_max](#), [dtprep](#)

**Examples**

```
#Get retention time and signal data
rtvec <- lcqc::simlc1[, "Time"]
sigvec <- lcqc::simlc1[, "Signal"]

#Get second derivatives
svec <- chrom_deriv(rtvec, sigvec)[[2]]

#Set locations of left and right inflection points
lvec <- c(527, 601, 621, 810, 974, 1822, 2483)
rvec <- c(546, 611, 638, 828, 994, 1850, 2522)

acc_inf(xvals = rtvec, yvals = sigvec, sd = svec, linfo = lvec, rinfo = rvec)
```

---

acc\_max

---

*Obtain accurate peak apex data via quadratic fitting*


---

**Description**

Evaluates the shape of peak tops and, if appropriate, quadratically fits a parabola to derive accurate retention time (and signal) data.

**Usage**

```
acc_max(inds = seq_along(xvals), xvals, yvals, maxes, linfo, rinfo, ptypes)
```

**Arguments**

inds	Optional numeric vector of data indices (by default, the sequence along retention time data is used, i.e. seq_along(xvals)).
xvals	Retention time (x-axis) data as a numeric vector.
yvals	Signal (y-axis) data as a numeric vector.
maxes	Indices of peak maxima (apices) as a numeric vector.
linfo, rinfo	Indices of <b>left</b> (linfo) and <b>right</b> (rinfo) inflection points (e.g. approximated by <a href="#">chrom_detect</a> ).
ptypes	A character vector of peak apex types. There are 4 possible values: baseline-resolved ("B"), fused ("F"), shoulder ("S"), and round ("R") peaks.

## Details

The function tests various conditions to pick the appropriate method of estimating accurate peak retention time (and signal) among a 5-point quadratic fit or a 3-point quadratic fit. No fit is carried out for shoulder and round peaks, or those where at least one inflection point is above the maximum in terms of signal. For others, provided the inflection point width is >5 points, a 5-point quadratic fit is attempted. Else, or if the estimated retention time is outside that of the 5 points used for the fit, a 3-point fit is instead attempted. If the estimated retention time is again outside that of the 3 points used for the fit, no fit is carried out.

## Value

A 3-column data.frame containing the accurate xy-coordinates and the utilized method for each peak (one of "5\_point", "3\_point", or "no\_fit").

## See Also

[dtprep](#), [acc\\_inf](#)

## Examples

```
#Get retention time and signal data
rtvec <- lcqc::simlc1[, "Time"]
sigvec <- lcqc::simlc1[, "Signal"]

#Set locations of left and right inflection points as well as apices
lvec <- c(527, 601, 621, 810, 974, 1822, 2483)
rvec <- c(546, 611, 638, 828, 994, 1850, 2522)
mvec <- c(536, 608, 628, 819, 983, 1834, 2499)
ptype <- c("F", "S", "F", "B", "B", "B", "B")

#Get accurate maxima
res <- acc_max(xvals = rtvec, yvals = sigvec, maxes = mvec,
  linfs = lvec, rinfs = rvec, ptypes = ptype)
```

---

addnms

*Add peak names to [chrom\\_detect](#) output*

---

## Description

A helper function that adds peak names to existing output of [chrom\\_detect](#).

## Usage

```
addnms(x, nms, whichpks = NA)
```

## Arguments

x	The output of <a href="#">chrom_detect</a> to add peak names to.
nms	A character vector of peak names. Must correspond to peak indices given in whichpks (if provided).
whichpks	An <b>optional</b> numeric vector of peak indices to include names for. Must be within the range of indices included in x. Defaults to NA, in which case nms must be provided for all peaks.

**Value**

A list of the same structure as that output by `chrom_detect` but including peak names.

**Examples**

```
## Not run:
dt <- lcqc::wf_detpeaks
pnms <- paste0("Ex", seq(7))
res <- addnms(dt, pnms)

## End(Not run)
```

---

blin\_als

---

*Iterative Asymmetric Least Squares (ALS) baseline correction*


---

**Description**

Asymmetric Least Squares (ALS) baseline correction, modified from package **baseline** for iterative convergence based on weighed Eilers smoothing (see also `smooth_what`).

**Usage**

```
blin_als(
  input,
  lambda = 6,
  p = 0.001,
  prec = 1e-08,
  maxit = 50,
  rm_neg = TRUE
)
```

**Arguments**

<code>input</code>	Numeric vector of signal to be baseline-corrected.
<code>lambda</code>	Second derivative constraint.
<code>p</code>	Weights of residuals (positive).
<code>prec</code>	Precision (error) tolerance required for baseline correction to reach convergence.
<code>maxit</code>	Maximum number of iterations (defaults to 50).
<code>rm_neg</code>	A TRUE/FALSE logical specifying whether points where the computed baseline is higher than the original signal should be discarded (TRUE by default).

**Value**

A list with 5 elements: `Original_Signal`, `Corrected_Signal`, `Baseline`, `Method` (always "als"), and `Parameters`.

**References**

Peng, J., Peng, S., Jiang, A., Wei, J., Li, C., Tan, J. (2010), 'Asymmetric Least Squares for Multiple Spectra Baseline Correction', *Analytica Chimica Acta* **683** (1), pp. 63-68, doi: <https://www.doi.org/10.1016/j.aca.2010.08>

**See Also**

[chrom\\_bline](#), [smooth\\_whit](#)

**Examples**

```
res <- blinc_als(lcqc::exgc1[, "Signal"])
blinc_plot(res)
```

---

blinc_chang	<i>Baseline correction (Chang's method)</i>
-------------	---

---

**Description**

Implementation of baseline correction according to Chang et al. (2007). Modified from package **TargetSearch**.

**Usage**

```
blinc_chang(
  input,
  threshold = 0.5,
  alpha = 0.95,
  bfrac = 0.2,
  segments = 100,
  sig_window = 10,
  fit = "linear",
  rm_neg = TRUE
)
```

**Arguments**

input	Input numeric vector containing signal to be baseline-corrected.
threshold	Position of the baseline relative to the noise component of the signal. Must be between 0 and 1.
alpha	High-pass filter parameter.
bfrac	Fraction of low-intensity fragments of the filtered signal that are assumed to be baseline.
segments	Number of segments to divide the filtered signal into.
sig_window	Signal window size (in points).
fit	Method used for baseline fitting. One of "linear" or the significantly slower "spline" (cubic).
rm_neg	A TRUE/FALSE logical specifying whether points where the computed baseline is higher than the original signal should be discarded (TRUE by default).

**Value**

A list with 5 elements: Original\_Signal, Corrected\_Signal, Baseline, Method (always "chang"), and Parameters.

## References

Chang, D., Banack, C.D., Shah, S.L. (2007), 'Robust baseline correction algorithm for signal dense NMR spectra', *Journal of Magnetic Resonance* **187**, pp. 288-292.

## See Also

[chrom\\_bline](#)

## Examples

```
res <- bline_chang(lcqc::exgc1[, "Signal"])
bline_plot(res)
```

---

bline_isrea	<i>Iterative Smoothing-Splines with Root Error Adjustment (ISREA) baseline correction</i>
-------------	---

---

## Description

Baseline correction that uses smoothing splines to estimate the baseline. Developed by Xu et al. (2021).

## Usage

```
bline_isrea(input, eta = 10, maxit = 100, rm_neg = TRUE)
```

## Arguments

input	Input numeric vector containing signal to be baseline-corrected.
eta	Convergence criterion. May take values from 0.0001 to 10 (based on Xu et al., 2021). Usually does not appreciably affect the results, but more computation time is needed for lower values.
maxit	Maximum number of iterations (defaults to 100).
rm_neg	A TRUE/FALSE logical specifying whether points where the computed baseline is higher than the original signal should be discarded (TRUE by default).

## Value

A list with 5 elements: Original\_Signal, Corrected\_Signal, Baseline, Method (always "isrea"), and Parameters.

## References

Xu, Y., Du, P., Senger, R., Robertson, J., Pirkle, J.L. (2021), 'ISREA: An Efficient Peak-Preserving Baseline Correction Algorithm for Raman Spectra', *Applied Spectroscopy* **75** (1), pp. 34-45, doi: <https://www.doi.org/10.1177/0003702820955245>.

## See Also

[chrom\\_bline](#)



**Examples**

```
res <- blinc_isrea(lcqc::exgc1[, "Signal"])
blinc_plot(res)
```

blinc\_plot

*Plot baseline correction results***Description**

Create a summary plot of baseline correction results containing the original signal, calculated baseline, and (optionally) baseline-corrected signal.

**Usage**

```
blinc_plot(
  input,
  xvar = "auto",
  xname = "Index",
  blinc = FALSE,
  corrsig = TRUE,
  asprat = 0.71
)
```

**Arguments**

input	Output from one of the baseline correction functions of <b>lcqc</b> : <a href="#">blinc_als</a> , <a href="#">blinc_chang</a> , <a href="#">blinc_poly</a> , <a href="#">blinc_isrea</a> , or <a href="#">chrom_blinc</a> .
xvar	An optional numeric vector of x values corresponding to the signal in input. If set to "auto", uses point indices instead.
xname	Title of x-axis (defaults to "Index").
blinc	A logical. Should the calculated baseline be included in the plot? Defaults to FALSE.
corrsig	A logical. Should baseline-corrected signal be included in the plot? Defaults to TRUE.
asprat	A numeric value setting the plot aspect ratio.

**Value**

A ggplot-class object containing the plot.

**See Also**

[chrom\\_blinc](#)

**Examples**

```
blinc_res <- chrom_blinc(lcqc::exgc1[, "Signal"], method = "als", slnt = TRUE)
blinc_plot(blinc_res, blinc = TRUE, corrsig = TRUE)
```

---

blines_poly	<i>Modified Polynomial Fit (ModPolyFit) baseline correction</i>
-------------	---

---

## Description

Iterative baseline correction algorithm based on automated polynomial fitting developed by Lieber & Mahadevan-Jansen (2003). Modified from package **baseline**.

## Usage

```
blines_poly(input, deg = 4, prec = 0.001, maxit = 100, rm_neg = TRUE)
```

## Arguments

input	Input numeric vector containing signal to be baseline-corrected.
deg	Degree of the polynomial fit.
prec	Precision (error) tolerance required for baseline correction to reach convergence.
maxit	Maximum number of iterations (defaults to 100).
rm_neg	A TRUE/FALSE logical specifying whether points where the computed baseline is higher than the original signal should be discarded (TRUE by default).

## Value

A list with 5 elements: Original\_Signal, Corrected\_Signal, Baseline, Method (always "poly"), and Parameters.

## References

Lieber, C.A., Mahadevan-Jansen, A. (2003), 'Automated Method for Subtraction of Fluorescence from Biological Raman Spectra', *Applied Spectroscopy* **57** (11), pp. 1363-1367, doi: <https://www.doi.org/10.1366/00037>

## See Also

[chrom\\_blines](#)

## Examples

```
res <- blines_poly(lcqc::exgc1[, "Signal"])
blines_plot(res)
```

browse\_visc

*Estimate dynamic viscosity of common HPLC mobile phases*

## Description

This function estimates the temperature-dependent dynamic viscosity  $\eta$  (in mPas) for pure organic solvents and their mixtures commonly used as mobile phases for High-Performance Liquid Chromatography (HPLC). It is also possible to estimate viscosities of aqueous mobile phases containing methanol (MeOH) or acetonitrile (MeCN) based on bilinear interpolation of experimental data collected from various sources. See **Details** for further information.

## Usage

```
browse_visc()
```

```
chrom_visc(ids, fracs, frac_type = "vol", temp = 25)
```

## Arguments

ids	A character vector of mobile phase component IDs to calculate viscosity for. One or more of: butanol ("buoh"), isopropanol ("ipa"), acetone ("acet"), acetonitrile ("mecn"), benzene ("benz"), chloroform ("chc13"), cyclohexane ("chex"), diethyl ether ("dee"), ethanol ("etoh"), ethyl acetate ("etac"), methanol ("meoh"), tetrahydrofuran ("thf"), and water ("h2o"). If the latter is included, ids must be of length 2 with the only other possible components being acetonitrile ("mecn") or methanol ("meoh"). See <b>Details</b> .
fracs	A numeric vector of mobile phase component fractions of equal length to ids.
frac_type	The <b>type</b> of fraction given in fracs. One of: volumetric ("vol", default), mass ("mass"), or mole ("mol").
temp	Temperature to calculate viscosity at. See <b>Details</b> and <a href="#">browse_visc</a> for acceptable temperature ranges for each solvent.

## Details

Dynamic viscosity and density of various organic solvents is calculated using data taken from the Dortmund Data Bank (DDB, 2024) and packaged with **lcqc**. Specifically, dynamic viscosity  $\eta$  (mPas) is calculated using the Vogel Equation (García-Colín et al., 1989), which uses experimentally derived empirical coefficients  $A_V$ ,  $B_V$ , and  $C_V$ . The result is also dependent on temperature, given as  $T_K$  in Kelvin. A full list of coefficients and the temperature ranges (in °C) within which they are valid is available upon calling [browse\\_visc](#).

$$\eta \text{ (mPas)} = \exp[A_V + (B_V / (C_V + T_K))]$$

Similarly, density  $\rho$  (kg m<sup>-3</sup>) is calculated via the DIPPR105 equation, which incorporates 4 empirical coefficients ( $A$ ,  $B$ ,  $C$ , and  $D$ ) alongside temperature (DDB, 2024; Silva et al., 2018).

$$\rho \text{ (kg m}^{-3}\text{)} = A / B^{1 + (1 - T_K / C)^D}$$

When a mixture of components is provided in ids, they have to be accompanied by mole ("mol"), mass ("mass"), or volumetric ("vol", default) fractions given in fracs. Whichever fraction type (frac\_type) is chosen, the others are calculated and included in function output. For example,

conversion to  $F_{mass}$  and  $F_{mol}$  from  $F_{vol}$  is carried out using density  $\rho$  and relative molecular mass ( $MM$ , g/mol) as follows for each component.

$$F_{mass} = F_{vol} \times \rho$$

$$F_{mol} = F_{mass}/MM$$

The overall viscosity  $\eta_{total}$  of the mixtures is calculated using the Linear Blend Rule **for purely organic mixtures**. Briefly, the contributing **mole** fraction ( $F_{mol}$ ) of each component  $i$  is multiplied by the corresponding viscosity value of the pure component, and results for all components obtained in this manner are summed.

$$\eta_{total} = \sum F_{mol(i)} * \eta_i$$

For **organic-water mixtures**, the Linear Blend Rule is not applicable due to the strong interactions between water molecules (Snyder et al., 1997). For this reason, **lcqc** is limited to estimating the viscosity of such mixtures via **bilinear interpolation** of experimental data obtained for the two most popular aqueous mobile phases in HPLC, **Methanol-Water** and **Acetonitrile-Water**, from various sources (Snyder et al., 2009; Thompson et al., 2006; Teutenberg et al., 2009; Wohlfarth & Wohlfahrt, 2001).

## Value

A named list containing results, a character string of various information about them, and the function call. The results element is a list named according to the mobile phase components specified in argument `ids` and containing a named numeric vector with the viscosity value ("`visc_mPas`"), temperature taken from `temp` ("`temp_degC`"), as well as mole, mass, and volume fractions of each component (suffixes `_molfrac`, `_massfrac`, and `_volfrac`, respectively).

## References

- Dortmund Data Bank (2024), 'Online Calculation', available at: <https://www.ddbst.com/calculation.html> (accessed 25.04.2024).
- García-Colín, L.S., del Castillo, L.F., Goldstein, P. (1989), 'Theoretical Basis for the Vogel-Fulcher-Tammann Equation', *Physical Review B* **40** (10), pp. 7040-7044, DOI: <https://www.doi.org/10.1103/PhysRevB.40.7040>.
- Silva, M., Vieira, B., Ottens, M. (2018), 'Preferential crystallization for the purification of similar hydrophobic polyphenols', *Journal of Chemical Technology & Biotechnology* **93** (7), pp. 1997-2010, DOI: <https://doi.org/10.1002/jctb.5526>.
- Snyder, L.R., Kirkland, J.J., Glajch, J.L. (1997), 'Appendix II: Properties of Solvent Used in HPLC', In: *Practical HPLC Method Development, Second Edition*, John Wiley & Sons, USA.
- Snyder, L.R., Kirkland, J.J., Dolan, J.W. (2009), 'Appendix I: Properties of HPLC Solvents', In: *Introduction to Modern Liquid Chromatography*, pp. 879-886, DOI: <https://doi.org/10.1002/9780470508183.app1>.
- Teutenberg, T., Wiese, S., Wagner, P., Gmehling, J. (2009), 'High-temperature liquid chromatography. Part II: Determination of the viscosities of binary solvent mixtures - Implications for liquid chromatographic separations', *Journal of Chromatography A* **1216** (48), pp. 8470-8479, DOI: <https://doi.org/10.1016/j.chroma.2009.09.075>.
- Thompson, J.W., Kaiser, T.J., Jorgenson, J.W. (2006), 'Viscosity measurements of methanol-water and acetonitrile-water mixtures at pressures up to 3500 bar using a novel capillary time-of-flight viscometer', *Journal of Chromatography A* **1134** (1), pp. 201-209, DOI: <https://doi.org/10.1016/j.chroma.2006.09.006>.

Wohlfarth, C., Wohlfahrt, B. (2001), '3 Mixtures of Water and Organic Compounds', In: *Landolt-Börnstein - Group IV Physical Chemistry, Volume 18A: Pure Organometallic and Organononmetallic Liquids, Binary Liquid Mixtures*, Springer-Verlag Berlin Heidelberg, DOI: [https://www.doi.org/10.1007/10639275\\_6](https://www.doi.org/10.1007/10639275_6).

## See Also

[chrom\\_tplate](#)

## Examples

```
#Pure component (MeOH)
vpure <- chrom_visc(ids = "meoh", fracs = 1, frac_type = "vol", temp = 25)

#Mixture (3:7 MeOH:EtAc)
vmix <- chrom_visc(ids = c("meoh", "etac"), fracs = c(0.3, 0.7), frac_type = "vol", temp = 25)

#Aqueous mobile phase (2:8 MeOH:H2O)
vaq <- chrom_visc(ids = c("meoh", "h2o"), fracs = c(0.2, 0.8), frac_type = "vol", temp = 25)
```

---

chk\_pack

*Check for and install missing packages*

---

## Description

Checks a character vector of package names, detects any CRAN packages not currently installed, installs, and loads them.

## Usage

```
chk_pack(packlist)
```

## Arguments

packlist	A list of packages to check for. Packages not found are automatically installed and called.
----------	---

## Value

Nothing.

## Examples

```
## Not run:
chk_pack(c("ggplot2", "knitr"))

## End(Not run)
```

chrom\_addmets

*Calculate additional HPLC column-specific performance metrics*

## Description

This function uses various column characteristics such as length, particle size, flow rate, internal diameter (among others) to calculate various column performance indicators. See **Details** for further information.

## Usage

```
chrom_addmets(
  which_mets = "all",
  t0,
  len,
  flow = NA,
  id = NA,
  deltap = NA,
  visc = NA,
  dp = NA
)
```

## Arguments

which_mets	A character vector of metrics to calculate. One or more of: "all" (default), "linvel" (Linear Velocity; mm/s), "porosity" (Packing Porosity; dimensionless), "flowres" (Flow Resistance; dimensionless), "pabil" (Permeability; mm <sup>2</sup> /s/bar), and/or "spabil" (Specific Permeability; mm <sup>2</sup> ).
t0	The numeric value for dead time (i.e. breakthrough time) in <b>minutes</b> .
len	The numeric column length (in mm).
flow	Flow rate (mL/min, numeric).
id	Column internal diameter (in mm, numeric).
deltap	Back pressure (bar; numeric).
visc	Dynamic viscosity of the mobile phase (mPas; numeric). May be calculated via <a href="#">chrom_visc</a> .
dp	Stationary phase particle size (in μm).

## Details

The linear velocity  $\mu$  of the mobile phase is simply related to column length  $L_c$  and true breakthrough time  $t_0$ .

$$\mu = L_c / t_0$$

It may be beneficial to verify breakthrough time in this equation, i.e. determine whether the non-retained analyte is really passing through the column at the velocity of the mobile phase, by calculating packing porosity  $\epsilon$ . The packing porosity of **bonded** silica-based stationary phases is *ca.* 0.65. Using this value as reference,  $\epsilon$  may be calculated simply from  $t_0$  (s),  $L_c$  (mm), flow rate  $F$  (mL min<sup>-1</sup>), and inner column diameter  $d_c^2$  (mm<sup>2</sup>).

$$\epsilon = 21 \times [(F t_0) / (d_c^2 L_c)]$$

Significantly higher or lower values of  $\epsilon$  ( $>1$  and  $<0.5$ , respectively) indicate retention of the analyte or its exclusion from the pores of the stationary phase (Meyer, 2010). Another useful metric is permeability  $K$ , which incorporates back pressure  $\Delta\rho$  (bar). A large value of  $K$  indicates poor column packing, while the reverse is characteristic of a leak. Values of  $15\text{ mm}^2\text{ s}^{-1}\text{ bar}^{-1}$  are typical for bonded phases at a fast flow rate of  $1.6\text{ mL min}^{-1}$ .

$$K = L_c^2/(\Delta\rho t_0)$$

Specific permeability  $K^\circ$  ( $\text{mm}^{-2}$ ) may also be calculated (typically  $4.0 \times 10^{-8}\text{ mm}^2$  for bonded phases) by incorporating flow rate, dynamic viscosity, column length, inner diameter, and back pressure.

$$K^\circ = 21 \times 10^{-8} \times [(F\eta L_c)/(d_c^2 \Delta\rho)]$$

Finally,  $K$  may be represented as a dimensionless metric called Flow Resistance  $\Phi$ , which facilitates comparison of different columns. One of the additional required parameters is particle size  $d_p$  ( $\mu\text{m}$ ).

$$\Phi = 4.7 \times [(\Delta\rho d_p^2 d_c^2)/(L_c \eta F)]$$

. A typical  $\Phi$  value of 1000 is observed for packed HPLC columns. Significant upward and downward deviations (e.g.  $>2000$  or  $<500$ ) are indicative of a blockage or voids in the packing, respectively.

For calculation of theoretical plates and additional dimensionless metrics such as reduced plate height and Separation Impedance to assess column performance, see [chrom\\_tplate](#).

### Value

A named list of length 3 containing a data.frame of results, various information about them, and the function call.

### References

Meyer, V.R. (2010), *Practical High-Performance Liquid Chromatography*, John Wiley & Sons, Chichester, United Kingdom.

### See Also

[chrom\\_visc](#), [chrom\\_tplate](#)

### Examples

```
chrom_addmets(t0 = 1, len = 250, id = 3.2, flow = 1.6, deltap = 70, visc = 0.33, dp = 5)
```

---

chrom\_amplim

*Find amplitude limit for peak detection*

---

### Description

Detects a suitable signal amplitude limit for peak detection using a simple quantile, relative differences between quantiles, or z-scores (see [z\\_thres](#)).

### Usage

```
chrom_amplim(x, method = "diff", pars = 1)
```

## Arguments

x	Numeric vector of signal from which to derive an amplitude limit.
method	One of: "quant" (simple quantiles), "diff" (relative differences between quantiles), or "zscore" (z-scores).
pars	Parameters required for calculation depending on the chosen method: <b>For "quant"</b> The quantile probability to use as the amplitude limit. Given as a single numeric percentage between 0.001 and 50. <b>For "diff"</b> The percentage (between 0.001 and 50) of maximum difference between quantiles (calculated via a probability <code>seq(0, 1, 0.005)</code> ) used to locate the amplitude limit. <b>For "zscore"</b> A numeric vector of 3 parameters: the integer number of observations used to calculate moving average and standard deviation (lag in <code>z_thres</code> ), the numeric threshold (as a factor of moving standard deviation) at which to signal the presence of a peak (threshold in <code>z_thres</code> ), and numeric sensitivity of the amplitude limit to the average standard deviation throughout the signal (see <b>Details</b> ).

## Details

When method is "quant", the amplitude limit is simply determined as the quantile at the specified probability percentage between 0.001 and 50.

Method "diff" first calculates the quantile range of input vector x using `quantile(x, probs = c(0, 1, 0.005))`, derives the maximum difference between successive quantiles (via `max(diff())`), locates the earliest (lowest) quantile where a percentage of this difference between 0.001 and 50 (given in the pars argument) is exceeded, and uses this quantile as the amplitude limit.

Finally, method = "zscore" calculates **global mean values** of both the moving average  $\overline{MX}$  and the moving standard deviation  $\overline{SD}$  of the signal (derived via `z_thres`) and uses a sensitivity parameter *sens* (provided in `pars[3]`) to calculate the amplitude limit as follows:

$$AmpLim = \overline{MX} + (1/sens) \times \overline{SD}$$

Thus, *AmpLim* decreases with increasing *sens* (`pars[3]`).

## Value

A single numeric value of the amplitude limit

## See Also

[z\\_thres](#)

## Examples

```
sig <- lcqc::simlc1[, "Signal"]
lim1 <- chrom_amplim(sig, method = "quant", pars = 0.05)
lim2 <- chrom_amplim(sig, method = "diff", pars = 0.05)
lim3 <- chrom_amplim(sig, method = "zscore", pars = c(30, 5, 2))
lim3_alt <- chrom_amplim(sig, method = "zscore", pars = c(30, 5, 10))
```



chrom\_asym

*Calculate asymmetry metrics for chromatographic peaks***Description**

Calculates common asymmetry metrics for chromatographic data including the United States Pharmacopoeia (USP)/European Pharmacopoeia (EP) Tailing Factor ( $T_f$ ), and the Asymmetry Factor ( $A_s$ ). Additionally, the Total Peak Analysis (TPA) workflow proposed by Wahab et al. (2017) is also implemented for baseline-resolved peaks. See **Details** for further information.

**Usage**

```
chrom_asym(
  input,
  method = "all",
  which_peaks = "all",
  show_widths = TRUE,
  crit_w = "auto",
  asprat = 0.71,
  tpa_thres = 0.85,
  optmet = "nlp",
  plotset = "make"
)
```

**Arguments**

input	The output data from function <a href="#">chrom_detect</a> .
method	A character vector of method(s) to apply for asymmetry calculations. One or more of: "all" (default), "Tf" (USP $T_f$ ), "As" ( $A_s$ ), and/or "TPA" (Total Peak Analysis).
which_peaks	Selects specific peaks from input to process. Either "all" (default) or a numeric vector of peak indices included in input.
show_widths	A logical specifying whether the various peak half-widths at specific heights required for the calculations are included in the results (TRUE by default).
crit_w	The critical width parameter to use for baseline calculation via FastChrom ( <a href="#">fastchrom_bline</a> ). Defaults to "auto" or can be set manually as a numeric (usually equal to the minimum peak width at half height).
asprat	Aspect ratio of the plot(s). Defaults to 0.71.
tpa_thres	The numeric threshold between 0 and 0.99 to use for calculation of peak width and its relation to Gaussian standard deviation for TPA. Defaults to 0.85. See <b>Details</b> .
optmet	A character string specifying which method to use for iterative adjustment of the Gaussian model. One of "optim" or "nlp" (default).
plotset	A character string specifying whether results of TPA are visualized/shown via <a href="#">chrom_tpa</a> . One of "make" (generates plots without printing; default), "print" (generates and prints plots), or "none".

## Details

Peak asymmetry may be described in different ways, with most of the classical values providing a single numeric metric. All of the methods described below have been implemented in **lcqc**. Among the widespread metrics is the Asymmetry Factor  $A_s$ , calculated as the ratio between peak half-widths at 10% peak height on the **trailing** ( $B_{10}$ ) and **leading** ( $A_{10}$ ) edges of the peak. Thus, values greater than 1 indicate tailing, values less than one indicate fronting, and a tailing factor of exactly 1 is characteristic of a perfectly symmetrical peak.

$$A_s = B_{10}/A_{10}$$

Another metric commonly calculated is the USP/EP Tailing Factor  $T_f$ , which uses half-widths at 5% peak height.

$$T_f = \frac{(A_5 + B_5)}{2A_5}$$

Irrespective of whether  $T_f$  or  $A_s$  is used, acceptance criteria outlined in various pharmacopoeias list a range of **0.8-1.8** as acceptable. In practice, values of **0.9-1.2** are routinely achieved and desirable for typical test analytes. Both  $T_f$  and  $A_s$  provide information only about the **relative** amount of tailing or fronting, but fail to accommodate cases where both are present. For example, a widened peak with both tailing and fronting (a so-called *Eiffel Tower* effect) may have  $T_f$  and  $A_s$  values close to 1 despite being heavily distorted from a Gaussian profile. In order to provide separate measures of the absolute and relative contributions of tailing and fronting to the total peak shape, as well as to effectively visualize the results, **lcqc** also implements the Total Peak Analysis (TPA) workflow developed by Wahab et al. (2017) as a simple visual and quantitative assessment tool. First, the Gaussian standard deviation  $\sigma$  is estimated from a peak (with a maximum signal **normalized to unity**) using the peak width at a chosen percentage of peak height ( $W_H$ ):

$$\sigma = W_H / (2\sqrt{2\ln(1/H)})$$

The estimation of  $\sigma$  takes advantage of the fact that the tops of chromatographic peaks (after 80-85% peak height) usually follow a Gaussian distribution closely even for heavily-distorted peaks. Thus, a Gaussian model is constructed using the estimated  $\sigma$ , a peak maximum equal to 1, and the actual peak retention time. A linear constrained solver is then used to ensure the top 15% of the Gaussian model values are either equal to or enclosed by the actual chromatographic peak. Finally, absolute and relative (%) residuals are calculated separately for the leading and trailing edges of the peak, providing separate quantitative measures of the degree of peak fronting and tailing. Currently, the TPA procedure is limited by **lcqc** to only assess baseline-resolved peaks. Each peak is also assessed for its suitability for TPA based on residual sums to the left and right of the peak apex. The Gaussian model is supposed to be completely enclosed by the original chromatographic peak, but this is seldom the case, resulting in negative residuals where the model is outside the retention time boundaries of the peak. If **>50%** of residuals are found to be negative on either side of the peak, it is considered to be unsuitable for TPA.

## Value

A named list of length 4 containing the data.frame of results, the function call, the list of TPA plots (if any), and a character string providing various information about the results.

## References

- Meyer, V.R. (2010), *Practical High-Performance Liquid Chromatography*, John Wiley & Sons, Chichester, United Kingdom.
- The United States Pharmacopeial Convention (2021), *Physical Tests/<621> Chromatography* (USP 40-NF 35), available at: <https://www.usp.org/harmonization-standards/pdg/excipients/chromatography> (accessed 24.04.2024).

Wahab, M.F., Patel, D.C., Armstrong, D.W. (2017), 'Total Peak Shape Analysis: Detection and Quantitation of Concurrent Fronting, Tailing, and Their Effect on Asymmetry Measurements', *Journal of Chromatography A* **1509**, pp. 163-170, DOI: <https://doi.org/10.1016/j.chroma.2017.06.031>.

## See Also

[chrom\\_tpa](#), [chrom\\_detect](#)

## Examples

```
## Not run:
#Get data
dt <- lcqc::wf_detpeaks
res <- chrom_asym(dt, which_peaks = 3:7, show_widths = FALSE)

## End(Not run)
```

---

chrom\_bline

*Baseline correction using various methods*

---

## Description

This is the main baseline correction function able to apply several methods, such as Asymmetric Least Squares ([bline\\_als](#)), Modified Polynomial Fit ([bline\\_poly](#)), Chang's method ([bline\\_chang](#)), and ISREA ([bline\\_isrea](#)). Optionally, the results may be summarised visually using [bline\\_plot](#).

## Usage

```
chrom_bline(
  input,
  method = "als",
  pars = "default",
  plotres = "print",
  asprat = 0.71,
  rm_neg = TRUE,
  slnt = FALSE
)
```

## Arguments

input	Input numeric vector containing signal to be baseline-corrected.
method	Baseline correction method. One of: "als", "chang", "poly", "isrea", or "none" (for convenience when incorporating function into flexible workflows).
pars	A <b>named</b> vector of parameters specific to each baseline correction method. Applies default parameters when set to "default" - these are listed in <b>Details</b> .
plotres	Should a summary plot of baseline correction be created and/or printed? One of: "none", "plot", or "print".
asprat	Numeric value of the summary plot aspect ratio (defaults to 0.71).

rm_neg	A TRUE/FALSE logical specifying whether points where the computed baseline is higher than the original signal should be discarded (TRUE by default).
slnt	A TRUE/FALSE logical. Displays information about progress and results when FALSE (default).

### Details

This function incorporates all 4 approaches to baseline correction included in **lcqc**, each requiring unique parameters to be passed to its respective function. These can be provided in argument `pars` as a vector, i.e. `c()`, or simply set to "default" to use sensible defaults. These are listed below:

**For ALS (function `blines_als`)** `c(lambda = 6, p = 0.001, prec = 1e-08, maxit = 50)`

**For Chang (function `blines_chang`)** `c(threshold = 0.5, alpha = 0.95, bfrac = 0.2, segments = 100, sig_window = 10, fit = "linear")`

**For ModPolyFit (function `blines_poly`)** `c(deg = 4, prec = 0.001, maxit = 100)`

**For ISREA (function `blines_isrea`)** `c(eta = 10, maxit = 100)`

### Value

A list containing usual baseline correction results (see `blines_poly`, for example) and, when `plotres != "none"`, a `Bline_Plot` element containing a summary plot.

### See Also

[blines\\_als](#), [blines\\_chang](#), [blines\\_poly](#), [blines\\_isrea](#), [blines\\_plot](#)

### Examples

```
chrom_bline(lcqc::exgc1[, "Signal"], method = "als", plotres = "plot")
```

---

chrom_deriv	<i>Calculate first and second derivatives</i>
-------------	---

---

### Description

Calculate first and second derivatives from x and y data.

### Usage

```
chrom_deriv(x, y)
```

### Arguments

x	Numeric vector of x values.
y	Numeric vector of y values equal in length to that of x.

### Value

A list of two numeric vectors equal to x in length, containing first (`[[1]]`) and second (`[[2]]`) derivatives.

## Examples

```
xvals <- lcqc::simlc5[, "Time"]
yvals <- lcqc::simlc5[, "Signal"]
chrom_deriv(xvals, yvals)
```

---

chrom\_derlims

Determine First and Second Derivative Peak Detection Thresholds

---

## Description

Calculation of inferior (low) and superior (high) first/second derivative thresholds to use for peak detection.

## Usage

```
chrom_derlims(
  x,
  y = NULL,
  fder,
  sder,
  method = "ncore",
  outs = "iqr",
  sens = c(3, 1)
)
```

## Arguments

x	Vector of peak <b>start</b> indices (when y is provided), or a 2-column data.frame containing peak start and end indices.
y	Either NULL (if x is a data.frame) or a vector of peak <b>end</b> indices equal in length to x (if the latter is also a vector).
fder	A vector of first derivatives derived from a signal.
sder	A vector of second derivatives derived from the same signal as fder.
method	Method used to calculate derivative-based peak detection thresholds. One of: "vaz" (Vaz et al., 2016), "zscore" (Z-score), or "ncore" (noise core, <b>Details</b> ).
outs	Outlier detection/removal method. One <b>or more</b> of: "none", "all", "iqr" (inter-quartile range), "quant" (quantile), or "sd" (standard deviation, see <b>Details</b> ).
sens	A vector of two numeric sensitivity values for fine adjustment of peak detection thresholds. Unique for each method - see <b>Details</b> .

## Details

As a first step, the function removes all fder and sder points within the peak regions provided in x and/or y. Outliers are then removed from the resulting vector of noise using **one or more** (or none) of the following criteria in any combination:

"iqr" All values outside of  $Q_1 - 1.5 \times IQR$  and  $Q_3 + 1.5 \times IQR$  are removed (where  $Q_1$ ,  $Q_3$ , and  $IQR$  are the first quartile, third quartile, and inter-quartile range, respectively).

"quant" All values outside of the 2.5% and 97.5% quantiles are removed.

"sd" All values outside  $\bar{X} \pm 2.24 \times SD$  are removed (where  $\bar{x}$  and  $SD$  are the mean and standard deviation of the derivatives, respectively).

Inferior and superior (i.e. low and high) peak detection thresholds are then calculated from the remaining data using one of three methods. **Method "vaz"** is based on the work of Vaz et al. (2016). Differences between the global median of derivative signal  $M_D$  and the derivative signal are calculated and a new median  $M_{new}$  calculated from these results. Finally, the inferior and superior thresholds  $T$  are calculated as follows:

$$T = M_D \pm sens1 \times M_{new}/sens2$$

Here,  $sens1$  (given in `sens[1]`) is an empirical factor with which the threshold range widens/increases. Conversely,  $sens2$  (given in `sens[2]`) is inversely related to the threshold range.

**Method "zscore"** is a slight variation reliant solely on the median of derivatives:

$$T = M_D \pm sens1 \times M_D/sens2$$

Finally, **method "ncore"** follows the widely-practised noise-core approach (e.g. Waters Corporation, 2016). The noise core is determined as the standard deviation  $SD$  distance from the mean  $\bar{X}$  of derivatives:

$$T = \bar{X} \pm sens1 \times SD/sens2$$

Many industry-standard approaches recommend a value of 4 for  $sens1$ , stating that  $\pm 4SD$  best defines chromatographic noise.

## Value

A list with 3 elements: `$FD` and `$SD` each containing a vector of length 2 with inferior and superior first and second derivative thresholds, respectively. Finally, element `$noise` contains a list of `fder` (`$FD_noise`) and `sder` (`$SD_noise`) values determined to be noise.

## References

Vaz, F.A.S., Neves, L.N.O., Marques, R., Sato, R.T., Oliveira, M.A.L. (2016), 'Chromophoreasy, an Excel-Based Program for Detection and Integration of Peaks from Chromatographic and Electromigration Techniques', *Journal of the Brazilian Chemical Society* **27** (10), pp. 1899-1911.

Waters Corporation (2016), 'ApexTrack Integration: Theory and Application', document number 72000494EN, available at: [https://www.waters.com/waters/library.htm?cid=511436&lid=1546221&locale=en\\_GB](https://www.waters.com/waters/library.htm?cid=511436&lid=1546221&locale=en_GB) (accessed 17.04.2024).

## See Also

[z\\_thres](#), [chrom\\_deriv](#), [noise\\_plot](#)

## Examples

```
#Calculate derivatives
xvals <- lcqc::simlc1[, "Time"]
yvals <- lcqc::simlc1[, "Signal"]
ders <- chrom_deriv(xvals, yvals)
fder <- ders[[1]]
sder <- ders[[2]]

#Get optimal z-score peak regions
```

```

zlims <- z_optim(yvals)[[1]]

#Get derivative peak detection thresholds
thres <- chrom_derlims(x = zlims, y = NULL, fder, sder, method = "ncore", sens = c(2,1))
noise_plot(thres[["noise"]][["FD_noise"]], thres[["FD"]])
noise_plot(thres[["noise"]][["SD_noise"]], thres[["SD"]])

```

chrom\_detect

*Automatic detection and classification of chromatographic peaks*

## Description

This function uses a combination of various custom and industry-standard algorithms to automatically detect, filter, and classify chromatographic peaks using 1-dimensional data (only retention time and signal). This is the most complex function in **lcqc**.

## Usage

```

chrom_detect(
  chrom,
  vars = c("Time", "Signal"),
  trange = range(chrom[, vars[1]]),
  det_bunch = TRUE,
  pnms = NA,
  bline_method = "none",
  bline_pars = "default",
  smooth = rep("tri", 2),
  mpts = c(3, 3),
  crosspts = c(2, 2),
  ma_pts = c("auto", 7),
  ma_passes = c("auto", 1),
  sens = c(2, 1, 3),
  amp_thres = "zscore",
  ampfrac = 0.05,
  der_thres = c("ncore", "iqr"),
  apex_pars = c(0, 0.5),
  rej = c(wd = "auto", pa = NA, sn = NA, ht = NA),
  asprat = 0.71,
  rej_logic = rep("OR", 2),
  zscore_pars = "default",
  plot_corr = FALSE,
  plotset = "make"
)

```

## Arguments

chrom	A data.frame containing the chromatogram. Must include x- and y-axis data variables as listed in vars. When NA, these are assumed to be contained within the first two columns of the data.frame.
vars	Variable names of chrom corresponding to the retention time (x-axis) and signal (y-axis), in that order. Defaults to c("Time", "Signal").

trange	A numeric vector containing the retention time range of chrom to apply the workflow to. By default, the entire chromatogram is processed.
det_bunch	A logical switch. Should peak bunching be detected and mitigated? Defaults to TRUE.
pnms	<b>Optional</b> character vector of peak names. Defaults to NA. The number of peaks detected by the algorithm must be known before providing this parameter. Thus, a "practice run" is recommended before any names are set. Names may also be added to existing chrom_detect output via the helper function <a href="#">addnms</a> .
blin_method	Baseline correction method to use (if any). Defaults to "none". Possible methods are: "als", "chang", "poly", or "isrea". See <a href="#">chrom_bline</a> for details.
blin_pars	Parameters of the baseline correction. Set to "default" (sensible defaults. Otherwise, a <b>named</b> numeric vector of parameters unique to the chosen baseline correction method (blin_method). See <a href="#">chrom_bline</a> for further details.
smooth	A character vector of length 2 specifying the smoothing method to use for signal and first/second derivatives, respectively. Defaults to rep("tri", 2) (triangular smoothing for both). A single character value may also be provided and the same method is then applied to both signal and derivatives. For further details, see <a href="#">chrom_smooth</a> .
mpts	A numeric vector of length 2 specifying the number of points to survey on either side to confirm (or reject) detected signal ( <b>first element</b> ) and derivative ( <b>second element</b> ) extremes (i.e. maxima and minima). A single numeric value may also be provided and will then be used for both data types. Defaults to c(3, 3).
crosspts	A numeric vector of length 2 specifying the number of points to survey on either side to confirm (or reject) detected zero crossings (downcrosses and upcrosses) for signal ( <b>first element</b> ) and derivatives ( <b>second element</b> ).
ma_pts, ma_passes	The numeric vector of length 2 specifying the number of smoothing <b>points</b> (ma_pts) and <b>passes</b> (ma_passes) to use to calculate smoothed signal and derivatives using the method(s) given in smooth. The first element may also be set to "auto", in which case ma_pts and/or ma_passes will be <b>automatically</b> determined via <a href="#">chrom_width</a> . The second element must then be a <b>number</b> specifying the starting points/passess to use in <a href="#">chrom_width</a> . The defaults for ma_pts and ma_passes are c("auto", 7) and c("auto", 1), respectively.
sens	A numeric vector of sensitivity parameters used for fine-tuning the first and second derivative peak detection thresholds via function <a href="#">chrom_derlims</a> (first two elements, respectively), and the amplitude limit obtained via <a href="#">chrom_amplim</a> (when amp_thres includes "zscore"). Defaults to c(2, 1, 3).
amp_thres	A character value/vector specifying method(s) to use for determination of the amplitude limit for peak detection. <b>One or more</b> of: "quant", "diff", or "zscore" (see <a href="#">chrom_amplim</a> ) for details).
ampfrac	A parameter for <a href="#">chrom_amplim</a> that helps determine the amplitude threshold via method(s) specified in amp_thres.
der_thres	A character vector of length 2 specifying the methods to use for determination of derivative-based peak detection thresholds via <a href="#">chrom_derlims</a> . The <b>first</b> element must be one of "vaz", "zscore", or "ncore", while possible values for the <b>second</b> element are "none", "all", "iqr", "quant", or "sd".
apex_pars	The <b>liftoff</b> and <b>touchdown</b> parameters to use for ApexTrack baseline expansion (see <b>Details</b> , <b>References</b> , and <a href="#">apexbnds</a> for details).



rej	<p>A <b>named</b> numeric vector specifying peak filters to apply for selective peak removal (carried out in two stages, before and after boundary detection and classification). Possible names are:</p> <ol style="list-style-type: none"> <li>1. For <b>pre</b>-classification (filtering round 1): minimum height ("ht"), S/N ratio ("sn").</li> <li>2. For <b>post</b>-classification (filtering round 2): inflection point width ("wd") and peak area ("pa").</li> </ol> <p>All values default to NA (no filters are applied).</p>
asprat	Aspect ratio of plots (defaults to 0.71).
rej_logic	A character vector of length 2 specifying the logic to use for two rounds of peak filtering ( <b>pre</b> - and <b>post</b> -classification). Possible values are "OR" or "AND". Defaults to rep("OR", 2). A single value may also be provided, in which case the same logic will be applied for both rounds of peak filtering.
zscore_pars	Either "default" or a list of numeric vectors providing sequences of 3 parameters to use for <code>z_optim</code> .
plot_corr	A logical switch. Should the baseline-corrected signal be plotted? When FALSE (default), the original signal is plotted instead.
plotset	A character string specifying whether data is visualized/shown. One of "make" (generates plots without printing; default), "print" (generates and prints plots), or "none".

## Value

A **list of lists** containing named elements results and plots. The latter includes 3 ggplot-class plot objects including the chromatogram with detected peak markers as well as both first and second derivative-based noise plots. Element results contains the following elements in a list:

**Chromatogram** The chromatogram data.frame containing retention times, signal, baseline-corrected and/or smoothed signals (if any), original and (if any) smoothed first/second derivatives, and peak ID assignments.

**Derivative\_Noise** A 2-column data.frame containing the first and second derivative noise from `chrom_derlims`.

**Peaks** A list of data.frame objects, each containing data for an individual detected peak. This includes retention time, original and (if any) smoothed signals, and associated first/second derivatives.

**Peak\_Extents** A data.frame containing key information about the detected peaks as detailed in `peakfind` (such as peak types, boundary types, inflection and upslope points etc.).

**Amplitude\_Limit** The numeric peak detection amplitude limit.

**Derivative\_Limits** A list with lower and upper peak detection thresholds derived from first and second derivatives (elements FD and SD, respectively).

**Zscore\_Limits** A numeric vector of starting and ending retention times of peak regions derived from `z_optim`.

**information** A character string containing summary information about the results.

**call** The function call.

## References

Pirttilä, K., Balmgård, D., Rainer, J., Pettersson, C., Hedeland, M., Brunius, C. (2022), 'Comprehensive Peak Characterisation (CPC) in Untargeted LC-MS Analysis', *Metabolites* **12** (2), article 137, DOI: <https://www.doi.org/10.3390/metabo12020137>.

Waters Corporation (2017), 'Empower Software Data Acquisition and Processing Theory Guide', document 715005481 (Rev. A), available at: [https://support.waters.com/KB\\_Info/Empower\\_Breeze/WKB57375\\_Empower\\_3\\_-\\_How\\_to\\_acquire\\_and\\_process\\_data](https://support.waters.com/KB_Info/Empower_Breeze/WKB57375_Empower_3_-_How_to_acquire_and_process_data) (accessed 19.04.2024).

## See Also

This workflow uses a multitude of exported and **unexported** functions:

**Exported** [chrom\\_width](#), [chrom\\_smooth](#), [chrom\\_bline](#), [chrom\\_deriv](#), [z\\_optim](#), [chrom\\_amplim](#), [chrom\\_derlims](#)

**Unexported** [peakmark](#), [peakfind](#) and its constituent functions

## Examples

```
#LC chromatogram
reslc1 <- chrom_detect(lcqc::simlc1, vars = c("Time", "Signal"))

#GC chromatogram
resgc <- chrom_detect(lcqc::exgc2, vars = c("Time", "Signal"), det_bunch = FALSE)

#GC chromatogram with baseline correction
resgc2 <- chrom_detect(lcqc::exgc1, vars = c("Time", "Signal"),
  bline_method = "als", det_bunch = FALSE)
```

---

chrom\_export

*Compile and export LCQC data and visualizations*

---

## Description

Compiles all key data and visualizations obtainable via various **lcqc** functions and exports to disk.

## Usage

```
chrom_export(
  input_list,
  expath = getwd(),
  plotpars = "default",
  plot_format = "png"
)
```

## Arguments

**input\_list** A **named** list of data to be exported. Must contain **one or more** of the following elements:

**pks** Peak detection data output from [chrom\\_detect](#).

	<b>acctops</b> Accurate peak apices and inflection points obtained by running <a href="#">dtprep</a> on <a href="#">chrom_detect</a> output. <b>int</b> Traditional peak integration data output from <a href="#">chrom_skim</a> . <b>icf</b> Iterative Curve Fitting and integration data output from <a href="#">chrom_icf</a> . <b>visc</b> Dynamic viscosity data output from <a href="#">chrom_visc</a> . <b>tp</b> Theoretical plates and other metrics output from <a href="#">chrom_tplate</a> . <b>asym</b> Asymmetry metrics and Total Peak Analysis (TPA) data output from <a href="#">chrom_asym</a> . <b>rf</b> Retention factors output from <a href="#">chrom_ret</a> . <b>sf</b> Separation factors output from <a href="#">chrom_sepf</a> . <b>res</b> Peak resolution data output from <a href="#">chrom_res</a> . <b>cperf</b> Additional performance metrics output from <a href="#">chrom_addmets</a> .
expath	The character string of the <b>directory</b> in which to create a folder and export the results.
plotpars	A <b>named</b> numeric vector of plotting parameters. <b>Only relevant when</b> <code>plot_format = "png"</code> . One or more of "w" (width, in cm), "h" (height, in cm), "psize" (point size), and "res" (resolution). When set to "default", the set of parameters are <code>c(w = 10, h = 12, psize = 12, res = 300)</code> .
plot_format	A character specifying which format to export plots in. One of "png" or "pdf".

### Value

A timestamped folder is created inside `expath` containing the compiled results in a **LCQC\_Results.csv** file, and .png/.pdf visualizations contained in a separate **Plots** directory.

### See Also

[chrom\\_detect](#), [chrom\\_skim](#), [chrom\\_icf](#), [chrom\\_tplate](#), [chrom\\_asym](#), [chrom\\_ret](#), [chrom\\_sepf](#), [chrom\\_res](#), [chrom\\_addmets](#), [read\\_shim](#), [lcqc\\_render](#)

### Examples

```
## Not run:
#Export to the working directory (visualisations in .PNG)
chrom_export(input_list = list(pks = lcqc:::wf_detpeaks,
  acctops = dtprep(lcqc:::wf_detpeaks),
  int = lcqc:::wf_ints,
  icf = lcqc:::wf_icf,
  visc = lcqc:::wf_viscs,
  tp = lcqc:::wf_tplate,
  asym = lcqc:::wf_asyms,
  rf = lcqc:::wf_kfs,
  sf = lcqc:::wf_sfs,
  res = lcqc:::wf_rs,
  cperf = lcqc:::wf_perfmets))

## End(Not run)
```

chrom\_icf

*Non-Linear Least Squares Iterative Curve Fitting (ICF) of chromatographic data using various models*

## Description

The workflow uses data from peaks detected by [chrom\\_detect](#) to model chromatographic peaks by iteratively fitting simple Gaussian and modified Gaussian models of varying complexity. See **Details** for further information.

## Usage

```
chrom_icf(
  input,
  method = "all",
  crit_w = "auto",
  optmet = "all",
  reprs = c(emg = "emg1"),
  modres = FALSE,
  plotset = "make",
  asprat = 0.71
)
```

## Arguments

input	The output list of data from <a href="#">chrom_detect</a> .
method	The character vector of method(s) to use for curve fitting. One or more of several available models: simple Gaussian ("gs"), Exponentially-Modified Gaussian ("emg"), Exponential-Gaussian Hybrid ("egh"), Empirically-Transformed Gaussian ("etg"), or "all" (default value which applies all available models and picks the best-performing model individually for each peak or peak group).
crit_w	The critical width parameter used to calculate peak group baselines via <a href="#">fastchrom_bline</a> .
optmet	The character vector of method(s) to use for iterative optimization as outlined in <a href="#">optim</a> . One or more of: "Nelder-Mead", "BFGS", "L-BFGS-B", "SANN", or "all" (default).
reprs	A <b>named</b> character vector of model representations (where more than one is available). Names must be present in method. Currently available representations are: EMG ("emg1" or "emg2").
modres	A logical switch which determines whether baseline-resolved peaks are modeled (FALSE by default).
plotset	A character string specifying whether data is visualized/shown via <a href="#">icf_plot</a> . One of "make" (generates plots without printing; default), "print" (generates and prints plots), or "none".
asprat	Aspect ratio of the plot (defaults to 0.71).

## Details

Iterative curve fitting (ICF) fits several Gaussian and modified Gaussian models to either a single or a group of chromatographic peaks and minimizes the associated Root Mean Squared Error (RMSE) of the resulting model through iterative optimization. Thus, a model that resembles actual chromatographic signal as closely as possible within the constraints of the chosen model is obtained. ICF is especially useful as a peak deconvolution method for moderately or even completely fused peaks. The `chrom_icf` workflow currently implements 3 popular ICF models and offers active selection of the most appropriate model for each examined peak or a group of up to 5 fused peaks. The available models include a simple Gaussian model (GS), the Exponentially-Modified Gaussian (EMG), and the Exponential-Gaussian Hybrid (EGH) that is considered a simpler, more robust alternative to EMG with faster computation time.

The GS model is based on the following equation (Nikitas et al., 2001):

$$H_{gs} = H_m \times e^{-((t_R - t_m)/2\sigma)^2}$$

Where  $H_m$  and  $t_R$  are the signal and retention time at peak maximum, respectively,  $t_R$  is the current retention time, and  $\sigma$  is the standard deviation of the curve, effectively approximated as half of the peak width at inflection point height. The Gaussian model thus has only 3 parameters that require optimization and is therefore computationally cheap, but cannot accommodate fronting or tailing phenomena often observed in chromatographic data.

The EMG model (e.g. Li, 1995; Nikitas et al., 2001; Caballero et al., 2002; Kalambet et al., 2011) incorporates an exponential component into the Gaussian model and is by far the most popular approach to modeling chromatographic peaks with tailing and fronting:

$$H_{emg} = A \times e^{0.5 \times (\sigma/\tau)^2 - ((t_R - t_0)/\tau)} \times \mathcal{P}((t_R - t_0)/\sigma - \sigma/\tau)/\tau$$

Sometimes also written as (Li, 1997):

$$H_{emg} = A/2\tau \times e^{(\sigma^2/2\tau^2 + (t_0 - t_R)/\tau)} \times (1 + \text{erf}((t_R - t_0)/\sqrt{2}\sigma - \sigma/\sqrt{2}\tau))$$

Where new parameters  $A$  and  $\tau$  are, respectively, the peak area and the exponential time constant, which is negative for fronting peaks and positive for tailing peaks. The EMG model takes longer than GS or EGH to compute and appears to be less stable.

The EGH model (Lan & Jorgenson, 2001; Li, 2002) is a simplified empirical equation that also accommodates fronting and tailing peaks by including a truncated exponential component into the equation:

$$H_{egh} = H_m \times e^{-(t_{R2} - t_0)^2 / (2\sigma^2 + \tau \times (t_{R2} - t_0))}$$

Where  $t_{R2}$  are those retention times where  $(2\sigma^2 + \tau \times (t_R - t_0)) > 0$ . EGH converges significantly faster than EMG, but somewhat lacks flexibility for modeling fronting peaks (similarly to EMG).

Finally, the ETG model is unique in incorporating 6 parameters describing the **leading** ( $k_l$ ,  $\lambda_l$ , and  $\alpha$ ) and **trailing** ( $k_r$ ,  $\lambda_r$ , and  $\beta$ ) peak edges, respectively. Including the peak height  $H_m$ , a total of 7 parameters are optimized. Additionally, the function requires estimates of the left and right inflection point times ( $t_l$  and  $t_r$ ) for each peak, which remain constant and are not optimized:

$$H_{etg} = (2H_m e^{0.5}) / ((1 + \lambda_l e^{k_l(t_l - t)})^\alpha + (1 + \lambda_r e^{k_r(t - t_r)})^\beta - 1)$$

The ETG model offers a unique advantage of loose coupling between the descriptions of leading and trailing peak edges, which are only related by one peak amplitude parameter (derived from peak height). Additionally, despite the iterative optimization of 7 parameters, the fitting procedure converges rapidly (is not computationally expensive) - often more so than EMG.

All of the above functions are submitted for iterative optimization via a penalty function based on the Root Mean Squared Error (RMSE), which may be represented as follows:

$$RMSE = \sum \sqrt{(y_i - \hat{y}_i)^2 / n}$$

Where  $y_i$ ,  $\hat{y}_i$ , and  $n$  are the original signal, modeled curve, and sample size (number of points in the original data), respectively. This error metric is used to evaluate model performance and select the best-suited model among GS, EMG, EGH, and ETG during iterative optimization via the `optim` function.

## Value

A list of length 5 containing the following elements:

**main\_data** A `data.frame` containing comprehensive results of ICF. Columns include data indices ("ind"), retention time ("x"), original signal ("orig\_y"), that corrected for the **global** baseline ("corr\_y"; see also `chrom_bline`), the individual peak or peak group baseline obtained via the FastChrom algorithm ("bline"; see also `fastchrom_bline`), the final baseline-corrected signal ("y"), ICF model type used ("model\_type"), the summed model curve for all peaks within a group ("modsum\_y"), one column each for individual peak models (`paste0("peak_", peak_ID)`), and the baseline-resolved peak group ID ("group").

**integ\_res** A `data.frame` containing the integration results for each peak, including the type of model used ("model\_type"), the associated RMSE ("model\_rmse"), and the integrated peak area ("pa").

**information** A character string of various statements about the results.

**call** The function call.

**modplot** An (optional) list of `ggplot` objects containing `icf_plot` visualization(s) of results.

## References

- Kalambet, Y., Kozmin, Y., Mikhailova, K., Nagaev, I., Tikhonov, P. (2011), 'Reconstruction of Chromatographic Peaks using the Exponentially Modified Gaussian Function', *Journal of Chromometrics* **25** (7), pp. 352-356, DOI: <https://doi.org/10.1002/cem.1343>.
- Li (1995), 'A Simplified Exponentially Modified Gaussian Function for Modeling Chromatographic Peaks', *Journal of Chromatographic Science* **33**, pp. 568-572, DOI: <https://www.doi.org/10.1093/chromsci/33.10.568>.
- Li (1997), 'Development and Evaluation of Flexible Empirical Peak Functions for Processing Chromatographic Peaks', *Analytical Chemistry* **69**, pp. 4452-4462, DOI: <https://www.doi.org/10.1021/ac970481d>.
- Li (2002), 'Comparison of the Capability of Peak Function in Describing Real Chromatographic Peaks', *Journal of Chromatography A* **952** (1), pp. 63-70, DOI: [https://doi.org/10.1016/S0021-9673\(02\)00090-0](https://doi.org/10.1016/S0021-9673(02)00090-0).
- Lan, K., Jorgenson, J.W. (2001), 'A Hybrid of Exponential and Gaussian Functions as a Simple Model of Asymmetric Chromatographic Peaks', *Journal of Chromatography A* **915** (1), pp. 1-13, DOI: [https://doi.org/10.1016/S0021-9673\(01\)00594-5](https://doi.org/10.1016/S0021-9673(01)00594-5).
- Nikitas, P., Pappa-Louisi, A., Papageorgiou, A. (2001), 'On the Equations Describing Chromatographic Peaks and the Problem of the Deconvolution of Overlapped Peaks', *Journal of Chromatography A* **912** (1), pp. 13-29, DOI: [https://doi.org/10.1016/S0021-9673\(01\)00524-6](https://doi.org/10.1016/S0021-9673(01)00524-6).
- Wahab, M.F., Armstrong, D.W., Hellinghausen, G. (2019), 'The Progress Made in Peak Processing', *LCGC Supplements* **32** (5), pp. 22-28.

**See Also**

This workflow uses a multitude of exported and **unexported** functions:

**Exported** [icf\\_plot](#), [dtprep](#), [fastchrom\\_bline](#), [integ](#)

**Unexported** Model-building functions ([icf\\_GS](#), [icf\\_EMG](#), [icf\\_EGH](#), [icf\\_ETG](#)), error calculation functions ([gs\\_penalty](#), [emg\\_penalty](#), [egh\\_penalty](#), [etg\\_penalty](#)), [init\\_egh](#), [peak\\_hw](#)

**Examples**

```
## Not run:
#Get data
det_res <- lcqc::wf_detpeaks
chrom_icf(det_res, method = "egh", modres = TRUE)
chrom_icf(det_res, method = c("egh", "etg"), modres = FALSE)

## End(Not run)
```

---

chrom\_plot

---

*Plot chromatogram with or without peak markers*


---

**Description**

Plots a chromatogram with various possible levels of complexity, from a simple plot of the signal versus retention time to inclusion of peak markers (e.g. peak boundaries, inflection, upslope, and apex points) and the peak detection signal amplitude limit.

**Usage**

```
chrom_plot(
  chrom_df,
  ptab = NA,
  chrom_vars = "auto",
  id = "auto",
  apex = NA,
  inf = NA,
  ups = NA,
  bound = NA,
  hlt = NA,
  norm_chrom = TRUE,
  cols = "default",
  ptab_mode = "index",
  zoom = "auto",
  which_peaks = NA,
  lablim = NA,
  amp_line = NA,
  xlab = chrom_vars[1],
  ylab = chrom_vars[2],
  plot_title = "Chromatogram",
  draw = TRUE,
  asprat = 0.71
)
```

**Arguments**

chrom_df	A data.frame containing time (x-axis) and signal (y-axis) variables, whose colnames are included in chrom_vars.
ptab	The peak table containing peak apex/inflection/upslope/boundary point markers, provided as a data.frame. Often output from function <a href="#">chrom_detect</a> or <a href="#">peakfind</a> .
chrom_vars	A character vector of length 2 providing column names of x- and y-axis variables present in chrom_df.
id	Column name of the peak ID (index) included in ptab. If set to "auto" (default), peak indices are assigned automatically by seq(1, nrow(ptab), 1).
apex	Controls plotting of peak <b>apices</b> . Either a character vector of length 2 with column names (present in ptab) containing indices of left and right peak boundaries, or NA (default).
inf	Controls plotting of peak <b>inflection</b> points. Either a character vector of length 2 with column names (present in ptab) containing indices of left and right peak boundaries, or NA (default).
ups	Controls plotting of peak <b>upslope</b> points. Either a character vector of length 2 with column names (present in ptab) containing indices of left and right upslope points, or NA (default).
bound	Controls plotting of peak <b>start/end boundaries</b> . Either a character vector of length 2 with column names (present in ptab) containing indices of left and right peak boundaries, or NA (default).
hlt	A vector of peak apex <b>indices</b> present in ptab that must be highlighted.
norm_chrom	A logical indicating whether the chromatogram signal (y-axis) should be normalized to a maximum value of 100. Defaults to TRUE.
cols	Either "default" or a <b>named</b> vector of recognized colors. Accepted names are: "id" (peak number), "main" (main plot), "apex" (peak apices), "hlt" (highlight for apices set in hlt), "inf" (inflection points), "ups" (upslope points), "bound" (peak start/end boundaries), "ampline" (signal amplitude threshold for peak detection).
ptab_mode	A character value denoting whether peak marker data in ptab is provided as <b>data indices</b> ("index", default) or <b>real values</b> , e.g. retention time ("real").
zoom	Either a vector of length 2 providing c(low, high) x-axis limits, or set to "auto" to automatically zoom into the detected peak region.
which_peaks	Either NA (default) or a numeric vector of ptab peak indices to plot, discarding the rest.
lablim	Either NA (default) or a single numeric value between 0 and 100. Denotes the <b>relative</b> signal threshold below which peak apices are not labeled with numbers.
amp_line	Amplitude limit to be plotted as a dashed line, given as a single numeric value. Defaults to NA (no line plotted).
xlab	The x-axis character label.
ylab	The y-axis character label.
plot_title	The main title of the plot (character).
draw	Should the generated plot be shown in the graphics window? TRUE by default.
asprat	Aspect ratio of the plot (defaults to 0.71).



**Value**

A ggplot-class object containing the plot.

**Examples**

```
#Simple plot
chrom_plot(lcqc::simlc1)

#Plot with peak markers
## Not run:
dt <- lcqc::wf_detpeaks[["results"]]
chrom <- dt[["Chromatogram"]] #Get chromatogram data
pt <- dt[["Peak_Extents"]] #Get peak table
alim <- dt[["Amplitude_Limit"]] #Get amplitude limit
chrom_plot(chrom_df = chrom, ptab = pt, norm_chrom = TRUE, id = "peak", apex = "ind_finmax",
  inf = c("ind_linf", "ind_rinf"), ups = c("ind_lups", "ind_rups"),
  bound = c("ind_starts", "ind_ends"), amp_line = alim)

## End(Not run)
```

chrom\_res

*Calculate HPLC chromatographic resolution using various methods***Description**

Calculates chromatographic resolution using the full width, width at 50%, and separation factor methods (see **Details**).

**Usage**

```
chrom_res(
  input,
  peaks1 = "all",
  peaks2 = "all",
  method = "all",
  ks = c(1, "peak"),
  crit_w = "auto",
  verbose_res = FALSE
)
```

**Arguments**

input	The output list from <a href="#">chrom_detect</a> .
peaks1, peaks2	Either "all" (default) or numeric vectors of <b>start</b> and <b>end</b> peaks for which to calculate separation factors. Each element of peaks1 therefore corresponds to the same element of peaks2. When set to "all", resolution is calculated for pairs of successive peaks (e.g. 1:2, 2:3 etc.).
method	A character vector specifying the method(s) to use for the calculations. One or more of: "all" (default), "W0" (full width), "W50_1" (50% width, variant 1), "W50_2" (50% width, variant 2), and/or "sepret" (separation factor and theoretical plate method).

ks	Used to configure retention factor calculations. Either a numeric value of the dead time (in minutes), or a vector of length 2 specifying that dead time is to be obtained from the retention time of a specific "peak", and providing the numeric index of said peak that must be present in input data.
crit_w	The critical width parameter to use for baseline calculation via FastChrom ( <a href="#">fastchrom_bline</a> ). Defaults to "auto" or can be set manually as a numeric (usually equal to the minimum peak width at half height).
verbose_res	A logical switch specifying whether to include intermediate results such as retention factors and separation factors into the results. Defaults to FALSE.

## Details

Chromatographic resolution  $R$  may be calculated between two neighbouring peaks using various approaches which incorporate, in one form or another, information about both peak apices and widths. Only the peak apices are separated at  $R = 1$ , while near-complete resolution of similarly-sized peaks is achieved at  $R \geq 1.5$ . For peaks 1 and a **later-eluting** 2, the simpler and more widespread calculation methods involve the use of retention times  $t_{R1}$  and  $t_{R2}$ , and corresponding peak widths at either the base ( $W_1$  and  $W_2$ ) or 50% peak height ( $W_{0.5h1}$  and  $W_{0.5h2}$ ). Three different equations commonly utilized with these parameters are presented below. The first of these uses the full peak width at the base and is therefore the most conservative measure that is relatively more affected by non-Gaussian peak shapes, often resulting in lower values than those obtained by other methods.

$$R = (t_{R2} - t_{R1}) / (0.5 \times (W_1 + W_2))$$

$$R = 1.176 \times [(t_{R2} - t_{R1}) / (W_{0.5h1} + W_{0.5h2})]$$

$$R = (t_{R2} - t_{R1}) / [1.7 \times 0.5 \times (W_{0.5h1} + W_{0.5h2})]$$

**For isocratic separations**, where the number of theoretical plates between adjacent peaks should largely be the same, the value of  $R$  may also be related to peak retention factors ( $k_1$  and  $k_2$ ), separation factor  $\alpha$ , and average number of theoretical plates  $\bar{N}$  by the following equation (sometimes referred to as the fundamental resolution equation):

$$R = (\sqrt{\bar{N}}/4) \times [(\alpha - 1)/\alpha] \times [k_2/(1 + k_2)]$$

## Value

A named list of length 4 containing a data.frame of results, which includes the peak IDs (id1 and id2), peak types (type1 and type2), and resolution values calculated via one or more methods specified in method and prefixed with res\_. If verbose\_res is TRUE, additional data from [chrom\\_ret](#) and [chrom\\_sepf](#) are included. Also included is the dead time t0, a character string of various information about the results, and the function call as separate list elements.

## References

Meyer, V.R. (2010), *Practical High-Performance Liquid Chromatography*, John Wiley & Sons, Chichester, United Kingdom.

## See Also

[chrom\\_sepf](#), [chrom\\_ret](#), [chrom\\_detect](#), [fastchrom\\_bline](#)

## Examples

```
## Not run:
#Get data and run
dt <- lcqc::wf_detpeaks
res <- chrom_res(input = dt, ks = c(1,"peak"))

## End(Not run)
```

---

chrom\_ret

---

*Calculation of Retention Factors for HPLC analytes*


---

## Description

Calculates retention factors for chromatographic peaks based on column dead time. See **Details** for further information.

## Usage

```
chrom_ret(input, t0_mode = "peak", t0, peaks = "all", crit_w = "auto")
```

## Arguments

input	The output list from <a href="#">chrom_detect</a> .
t0_mode	The mode to use for calculation of dead time. One of: "peak" or "manual", where <i>t0</i> is taken directly as the time of specified peak maximum or given manually in minutes, respectively.
t0	Dead time of the column, i.e. breakthrough time. Either given in <b>minutes</b> if t0_mode is "manual", or given as a single peak index included in which_peaks.
peaks	Either "all" (default) or a numeric vector of peak indices to calculate retention factors for.
crit_w	The critical width parameter to use for baseline calculation via FastChrom ( <a href="#">fastchrom_bline</a> ). Defaults to "auto" or can be set manually as a numeric (usually equal to the minimum peak width at half height).

## Details

The retention factor *k* is used to gauge the relative retention of an analyte compared to that of an unretained solute (such as uracil for reversed-phase separations). It is calculated from dead time *t*<sub>0</sub>, a.k.a. breakthrough time, using the following simple equation:  $k = (t_R - t_0)/t_0$ . Typical RP separations include compounds with *k* ranging between 1 and 10 (up to 20 for difficult separations). In addition to providing a retention time manually, it is possible to set the dead time as the retention time of one of the peaks in **input** when t0\_mode is set to "peak".

## Value

A named list of length 4 containing a data.frame of results, which includes the peak id, peak type (see [chrom\\_detect](#) for descriptions of available peak types), retention time *rt*, the dead time *t*<sub>0</sub>, and the calculated retention factors *k*. **If** the input data (from [chrom\\_detect](#)) also includes the Compound names for each peak, these are included as the first column. Also included is the dead time *t*<sub>0</sub>, a character string of various information about the results, and the function call as separate list elements.

## References

Meyer, V.R. (2010), *Practical High-Performance Liquid Chromatography*, John Wiley & Sons, Chichester, United Kingdom.

## See Also

[chrom\\_detect](#), [fastchrom\\_bline](#)

## Examples

```
## Not run:
#Get data and run
dt <- lcqc::wf_detpeaks
res <- chrom_retf(input = dt, t0_mode = "peak", t0 = 1)

## End(Not run)
```

---

chrom\_sepf

*Calculation of Separation Factors for HPLC analytes*

---

## Description

Calculates the retention and separation factors for any number of peak pairs.

## Usage

```
chrom_sepf(
  input,
  peaks1 = "all",
  peaks2 = "all",
  ks = c(1, "peak"),
  crit_w = "auto"
)
```

## Arguments

input	The output list from <a href="#">chrom_detect</a> .
peaks1, peaks2	Either "all" (default) or numeric vectors of <b>start</b> and <b>end</b> peaks for which to calculate separation factors. Each element of peaks1 therefore corresponds to the same element of peaks2. When set to "all", separation factors are calculated for pairs of successive peaks (e.g. 1:2, 2:3 etc.).
ks	Used to configure retention factor calculations via <a href="#">chrom_retf</a> . Usually a vector of length 2 specifying the character mode of calculating retention factors (one of "peak" or "manual") and an accompanying numeric value as a parameter. When in "peak" mode, the accompanying value is the index of a peak present in input data whose retention time is used as dead time $t_0$ to calculate retention factors from. When mode is "manual", the accompanying value must simply be the dead time itself (in minutes). Alternatively, retention indices may be provided as a numeric vector of length equal to that of peaks1 and peaks2, in which case <a href="#">chrom_retf</a> is not called.

`crit_w` The critical width parameter to use for baseline calculation via FastChrom ([fastchrom\\_bline](#)). Defaults to "auto" or can be set manually as a numeric (usually equal to the minimum peak width at half height).

## Details

Separation factor  $\alpha$  (or *relative retention* in some older texts) is useful to determine the relative degree of separation between any two peaks in a chromatogram. It is therefore a measure of selectivity. For two peaks  $i$  and a **later-eluting**  $j$ ,  $\alpha$  can be calculated as a simple ratio of retention factors  $k_i$  and  $k_j$ . If  $\alpha \leq 1$ , no separation between the components takes place.

$$\alpha = k_j / k_i$$

## Value

A named list of length 4 containing a data.frame of results, which includes the peak IDs (id1 and id2), peak types (type1 and type2), retention factors k1 and k2, and the calculated separation factors sep\_factor. **If** the input data (from [chrom\\_detect](#)) also includes the Compound names for each peak pair, these are included as the first two columns. Also included is the dead time t0, a character string of various information about the results, and the function call as separate list elements.

## References

Meyer, V.R. (2010), *Practical High-Performance Liquid Chromatography*, John Wiley & Sons, Chichester, United Kingdom.

## See Also

[chrom\\_retf](#), [chrom\\_detect](#), [fastchrom\\_bline](#)

## Examples

```
## Not run:
#Get data and run
dt <- lcqc::wf_detpeaks
res <- chrom_sepf(input = dt, peaks1 = c(2:6), peaks2 = c(3:7), ks = c(1,"peak"))

## End(Not run)
```

---

chrom\_skim

*Traditional peak integration of chromatographic data*

---

## Description

Creates baseline-resolved, perpendicular drop, and/or various skimmed baselines (tangent/exponential/Gaussian) and integrates peaks detected via [chrom\\_detect](#) via the Trapezoidal Rule. Results may be optionally visualized. Also see **Details**.

## Usage

```
chrom_skim(
  input,
  method = "gskim",
  skim = 10,
  dyson = 10,
  crit_w = "auto",
  asprat = 0.71,
  plotset = "make"
)
```

## Arguments

input	The output object of function <a href="#">chrom_detect</a> .
method	The preferred integration/skimming method. One of: perpendicular drop ("pdrop"), tangent skim ("tskim"), exponential skim ("exskim"), or Gaussian skim ("gskim").
skim	The numeric Skim Ratio criterion to determine suitability of a parent-child peak pair for skimming. Defaults to 10 (see <b>Details</b> ).
dyson	The numeric Dyson criterion to determine suitability of a parent-child peak pair for skimming. Defaults to 10 (see <b>Details</b> ).
crit_w	The critical width parameter used to calculate peak group baselines via <a href="#">fastchrom_bline</a> .
asprat	Aspect ratio of the plot (plotted via <a href="#">integ_plot</a> ). Defaults to 0.71.
plotset	A character string specifying whether data is visualized/shown via <a href="#">integ_plot</a> . One of "make" (generates plots without printing; default), "print" (generates and prints plots), or "none".

## Details

The workflow assesses peaks based on their resolution. First, baselines for fully-resolved peaks are calculated via the FastChrom algorithm (see [fastchrom\\_bline](#) and Johnsen et al., 2013). Common baselines for groups of **unresolved** peaks are also derived in the same manner. Boundaries between each pair of unresolved peaks are then either separated by a simple Perpendicular Drop baseline when method = "pdrop". Perpendicular Drop tends to increasingly distort peak areas with decreasing resolution (see Asher et al., 2009). Alternatively, peaks boundaries are tested for suitability for one of three "skimmed" baseline approaches - tangent skim (method = "tskim"), exponential skim ("exskim"), or Gaussian skim ("gskim") - by assessing a series of conditions (Agilent Technologies, 2016; Water Corporation, 2017):

1. Is the peak boundary classified as either fused or a shoulder?
2. Is at least one peak within the assessed pair classified as fused or a shoulder?

If either of these initial conditions are false, no skim is attempted. Otherwise, additional conditions are evaluated:

1. Is the earlier-occurring peak the parent peak or a child peak (decided based on the height of peak maxima)? This determines whether to apply a **front** or a **tail** skim.
2. Is the apex of the child peak lower than the closest (along the x-axis) inflection point of the parent peak? If this conditions is false, **exponential or Gaussian skimming are not attempted**.
3. Is the Skim-Valley ratio (child peak height over that of the inter-peak boundary/valley) lower than the threshold set in skim?

4. Is the height ratio of the parent peak to the child peak higher than the criterion set in dyson? If either of the conditions dependent on skim and dyson are false, **skimming is not carried out**.
5. Is the outer boundary of the child peak higher than the inter-peak boundary? If true, **tangent skimming is not possible**.

These conditions also determine the identity of the parent and child peak as well as the appropriate skim type (**front** or **tail**) for each peak pair. Thus, if a boundary is determined to be suitable based on the above conditions, construction and optimization of a skimmed baseline of the type specified in method is attempted.

For **front** tangent skim off the parent peak, straight lines are drawn from the inter-peak boundary to each point between the beginning and the maximum of the child peak (which occurs earlier in this case). For a **tail** tangent skim, lines are instead drawn between the inter-peak boundary and each point between the maximum and end of the child peak (which is now the later-occurring peak). Lines whose y-values are >2% of the maximum child peak signal at any point are discarded. Among the rest, the line whose end-point is closest to the outer boundary of the child peak and where <40% of the values are within 1% of the corresponding chromatographic signal value is selected.

For exponential skim, the following equation is used to build an exponential curve extending from the inflection point of the parent peak closest to the inter-peak boundary towards either the start (for **front** skim) or end (for **tail** skim) of the child peak.

$$H_{ex} = H_0 \times \exp(-B \times (t_R - t_0)) + A \times t_R + C$$

Where  $H_{ex}$  is the exponential curve value,  $H_0$  is the height (signal) at the inter-peak boundary (e.g. valley),  $B$  is the exponential growth/decay function (negative for **tail** skim),  $A$  is the slope of the parent peak baseline,  $C$  is the baseline offset of the parent peak, and  $t_R$  along with  $t_0$  are retention times at  $H_b$  and the inter-peak boundary, respectively. The initial exponential curve is constructed with values of  $B$  and  $C$  both set to 0. The offset constant  $C$  is then determined by the difference between the result and the signal at the inter-peak boundary, and constant  $B$  of the exponential fit is optimized for minimum Euclidean Distance between the curve and original chromatographic signal **in the parent peak region** (spanning from the closest inflection point to the inter-peak boundary). Finally, the final exponential curve spanning from the inter-peak boundary to the outer boundary of the child peak is plotted using optimized constants  $B$  and  $C$ . This is used as the skimmed baseline.

For Gaussian skim, the following general form of the Gaussian curve is used to construct a model between the apex of the **parent** peak and the inter-peak boundary.

$$H_{gs} = H_p \times \exp\left(-\left(\frac{t_R - t_0}{\sigma}\right)^2\right)$$

Where  $H_{gs}$  is the Gaussian curve value,  $H_p$  is the parent peak maximum signal,  $t_0$  is the corresponding retention time,  $t_R$  is the retention time at  $H_{gs}$ , and  $\sigma$  is the standard deviation of the Gaussian curve (estimated here as the half-width of the peak at inflection points). The Gaussian curve is iteratively optimized until Euclidean Distance between the resulting curve and original parent peak is minimized. The final curve is then constructed using optimized parameters between the **parent** peak apex and the outer boundary of the **child** peak. The model is then checked for two conditions:

1. Is the lowest point of the Gaussian curve higher than 1% of the parent peak maximum?
2. Are any points of the curve in the **child** peak region above the original signal (i.e. does the curve cross the chromatogram)?

If either of the above is true, the Gaussian curve is rejected, no further skim is attempted, and a Perpendicular Dropline is instead constructed. Otherwise, the Gaussian curve is truncated from the

**parent** peak maximum until the point where the curve is of **consistently** lower signal (i.e. height) than the original chromatogram.

Once all the baselines are constructed, the Trapezoidal Rule is used to integrate all peaks and calculate their peak areas ( $PA$ ).

$$PA = \sum (x_{i+1} - x_i) \times (y_{i+1} + y_i) / 2$$

Results are optionally visualized via `integ_plot`.

## Value

A list containing various data in the following named elements:

**orig\_data** A data.frame containing the original data indices ("ind"), retention times ("x"), as well as the raw and baseline-corrected signal ("orig\_y" and "y", respectively).

**indiv\_bln** A data.frame containing perpendicular drop and/or skimmed baselines for each individual peak. The data includes indices ("ind"), retention time ("x"), baseline-corrected signal ("y"), peak group ID ("group"), those of parent and child peaks ("peak1" and "peak2"), integration type ("type"), and a composite ID unique for each baseline ("combo\_id").

**grp\_bln** A data.frame containing peak group baselines (one baseline per group). Data include group indices ("group"), data indices ("ind"), retention times ("x"), original and baseline-corrected signals ("orig\_y" and "y"), baselines ("bln"), and second derivatives ("sd").

**integ\_res** The **main results** data.frame containing, among other general data such as retention time and signal, the integrated peak areas ("pa") and results of testing for skim suitability via 7 conditions outlined in **Details** ("true\_conds" and "failed\_conds").

**max\_marks** A data.frame containing accurate peak maxima retention time and signals (see `acc_max`) among other general data such as retention time and signal.

**information** A character string of various statements about baseline calculation and integration results.

**call** The function call.

**plot** An **optional** element containing the ggplot visualisation of results (see `integ_plot`).

## References

- Agilent Technologies (2016), 'Agilent OpenLAB CDS Data Analysis Reference Guide', document M8410-90031, available at: <https://www.agilent.com/cs/library/usermanuals/public/DataAnalysisReferenceGuide.pdf>.
- Asher, B.J., D'Agostino, L.A., Way, J.D., Wong, C.S., Harynuk, J.J. (2009), 'Comparison of peak integration methods for the determination of enantiomeric fraction in environmental samples', *Chemosphere* **75** (8), pp. 1042-1048, DOI: <https://doi.org/10.1016/j.chemosphere.2009.01.041>.
- Dyson, N. (1998), *Chromatographic Integration Methods*, The Royal Society of Chemistry (RSC Chromatography Monographs series), Herefordshire, United Kingdom.
- Johnsen, L.G., Skov, T., Houlberg, U., Bro, R. (2013), 'An automated method for baseline correction, peak finding and peak grouping in chromatographic data', *Analyst* **138** (12), pp. 3502-3511, DOI: <https://www.doi.org/10.1039/C3AN36276K>.
- Kalambet, Y., Kozmin, Y., Samokhin, A. (2018), 'Comparison of integration rules in the case of very narrow peaks', *Chemometrics and Intelligent Laboratory Systems* **179**, pp. 22-30, DOI: <https://doi.org/10.1016/j.chemolab.2018.06.001>.



SERAS (2000), 'Standard Operating Procedure 1001: Chromatographic Peak Integration Procedures', U.S. EPA Contract EP-W-09-031, available at: <https://clu-in.org/download/ert/1001-r00.pdf> (accessed 22.04.2024).

Waters Corporation (2017), 'Empower Software Data Acquisition and Processing Theory Guide', document 715005481 (Rev. A), available at: [https://support.waters.com/KB\\_Inf/Empower\\_Breeze/WKB57375\\_Empower\\_3\\_-\\_How\\_to\\_acquire\\_and\\_process\\_data](https://support.waters.com/KB_Inf/Empower_Breeze/WKB57375_Empower_3_-_How_to_acquire_and_process_data)(accessed19.04.2024).

See Also

This workflow uses a multitude of exported and **un**exported functions:

**Exported** [dtprep](#), [fastchrom\\_bline](#), [integ](#), [integ\\_plot](#)

**Unexported** [chkdt](#), [skimming](#) functions ([tskim](#), [exskim](#), [gskim](#)) and their helper functions, [pdrop](#), [skimcomp](#)

Examples

```
## Not run:
#Get data and integrate via Perpendicular Drop
det_peaks <- lcqc:::wf_detpeaks
chrom_skim(det_peaks, method = "pdrop")

#Forced exponential skim
chrom_skim(det_peaks, method = "exskim", dyson = 1.5, skim = NA)

## End(Not run)
```

---

chrom_smooth	<i>Chromatogram smoothing</i>
--------------	-------------------------------

---

Description

Smoothes an equi-spaced signal using various common approaches, including rectangular, triangular, and Savitsky-Golay (S-G) smoothing.

Usage

```
chrom_smooth(x, method = "rect", pts = 3, passes = 3)
```

Arguments

x	Input numeric vector of equi-spaced values.
method	Smoothing method. One of: "rect" (rectangular), "tri" (triangular), "sg_quad" (S-G quadratic), or "sg_quart" (S-G quartic, see <b>Details</b> ).
pts	Number of points to use for smoothing (defaults to 3).
passes	Number of smoothing passes to apply (defaults to 3).

## Details

The function includes several smoothing approaches. The simplest method, "rect", is a *rectangular boxcar* algorithm, which replaces each points with the [mean](#) of several neighboring points. For example, for a 3-point smooth:

$$S_j = \frac{Y_{j-1} + Y_j + Y_{j+1}}{3}$$

Triangular smoothing ("tri") implements a weighted algorithm. For example, a 5-point smooth becomes:

$$S_j = \frac{Y_{j-2} + 2Y_{j-1} + 3Y_j + 2Y_{j+1} + Y_{j+2}}{9}$$

Finally, the Savitsky-Golay (S-G) smoothing methods, also known as digital smoothing polynomial filters, implement moving average-based smoothing via a quadratic ("sg\_quad") or quartic ("sg\_quart") polynomial regression. The latter usually performs better with narrower peaks, but may distort wider peaks.

## Value

The smoothed signal (as a numeric vector).

## References

O'Haver, T. (2023), *A Pragmatic Introduction to Signal Processing with Applications in Scientific Measurement*, independently published, Maryland, USA, available at: <https://terpconnect.umd.edu/~toh/spectrum/> (last accessed 16.04.2024).

Isnanto, R. Rizal (2011), 'Comparison of Several Smoothing Methods in Nonparametric Regression', *Jurnal Sistem Komputer* **1** (1), pp. 41-47.

## See Also

[chrom\\_width](#)

## Examples

```
chrom_smooth(lcqc::simlc5[, "Signal"], method = "tri")
```

---

chrom\_tpa

*Carry out Total Peak Analysis (TPA) for a selected chromatographic peak*

---

## Description

Calculates the Total Peak Analysis (TPA) metrics as described by Wahab et al. (2017) and optionally visualizes the results. See [chrom\\_asym](#) for further information.

**Usage**

```
chrom_tpa(
  input,
  vars = c("x", "y"),
  width = "auto",
  accmax = "auto",
  thres = 0.85,
  pknum = "",
  opt = "nlp",
  asprat = 0.71,
  print_plot = TRUE,
  plot_info = FALSE
)
```

**Arguments**

input	A data.frame containing retention time and signal data of the peak for which TPA is to be carried out.
vars	A character vector of length 2 containing the column names of retention time and signal, in that order.
width	Either "auto" (default) or a single numeric value of the peak width (in retention time units).
accmax	Either "auto" (default) or a numeric vector of length 2 containing the retention time and signal (in that order) of the peak for which TPA is to be carried out.
thres	The numeric threshold between 0 and 0.99 to use for calculation of peak width and its relation to Gaussian standard deviation for TPA. Defaults to 0.85.
pknum	A character string to append to the plot title. Usually contains information like the peak number and/or compound name.
opt	A character string specifying which method to use for iterative adjustment of the Gaussian model. One of "optim" or "nlp" (default).
asprat	Aspect ratio of the plot (defaults to 0.71).
print_plot	A logical switch. Should the generated plot be printed? TRUE by default.
plot_info	A logical switch. Should extra information about the suitability of the peak for TPA be included in the plot? FALSE by default.

**Value**

A named list containing the following elements:

**results** A named numeric vector of key results including the peak width at thres, the Gaussian standard deviation ("sigma"), the **absolute** residual sum ("resid\_sum"), its front- and back-of-peak components ("resid\_front" and "resid\_back"), corresponding percentage contributions to fronting and tailing ("percent\_fronting" and "percent\_tailing"), and a binary measure of the suitability of the peak for TPA ("tpa\_suitability") determined from the residuals. See [chrom\\_asym](#).

**Peak\_Data** A data.frame containing the retention time ("rt"), normalized peak signal ("normpeak"), its Gaussian model ("gausspeak"), residuals between the normalized peak and the Gaussian model ("resids"), and their absolute counterparts ("aresids").

**Plot** A ggplot object containing TPA visualization.

**See Also**

[chrom\\_asym](#), [peak\\_hw](#)

**Examples**

```
## Not run:
#Get data
dt <- lcqc::wf_detpeaks[["results"]][["Peaks"]][["peak_7"]]
chrom_tpa(dt, vars = c("Time", "Signal"), pknum = "7 (Placeholder)", plot_info = TRUE)

## End(Not run)
```

---

chrom_tplate	<i>Number of Theoretical Plates and related metrics for LC columns</i>
--------------	--

---

**Description**

Calculates the theoretical plate number  $N$ , Height Equivalent to a Theoretical Plate (HETP), reduced plate height  $h$ , and separation impedance  $E$  using various methods such as 5-sigma (S5), European/British Pharmacopoeia (EP), Area-Height (AH), and Exponentially-Modified Gaussian (EMG). See **Details** for further information.

**Usage**

```
chrom_tplate(
  input,
  method = "all",
  pa = NA,
  len = NA,
  dp = NA,
  show_widths = TRUE,
  crit_w = "auto",
  which_peaks = "all",
  deltap = NA,
  visc = NA,
  t0 = NA,
  t0_mode = "peak",
  imped_met = "all"
)
```

**Arguments**

input	The output of function <a href="#">chrom_detect</a> .
method	The method(s) to use for the calculation of $N$ . One or more of: full-width ("FW"), 5-sigma ("S5"), European Pharmacopoeia ("EP"), inflection point width ("inf"), Area-Height ("AH"), Exponentially-Modified Gaussian ("EMG"), or "all" (default).
pa	<b>Optional</b> numeric vector of peak areas of length equal to the examined number of peaks. Required if method includes "AH".

len	<b>Optional</b> numeric column length (in mm). Defaults to NA. Required for calculation of HETP.
dp	<b>Optional</b> numeric column stationary phase particle diameter (in $\mu\text{m}$ ). Required alongside len for calculation of reduced plate height $h$ .
show_widths	A logical specifying whether the various peak half-widths at specific heights required for the calculations are included in the results (TRUE by default).
crit_w	The critical width parameter to use for baseline calculation via FastChrom ( <a href="#">fastchrom_bline</a> ). Defaults to "auto" or can be set manually as a numeric (usually equal to the minimum peak width at half height).
which_peaks	Selects specific peaks from input to process. Either "all" (default) or a numeric vector of peak indices included in input.
deltap	<b>Optional</b> back pressure observed for the column with a specific mobile phase (in <b>bar</b> ). Required for calculation of separation impedance $E$ (see <b>Details</b> ).
visc	<b>Optional</b> dynamic viscosity of the mobile phase (in <b>mPas</b> ). May be calculated by <a href="#">chrom_visc</a> . Required for calculation of separation impedance $E$ (see <b>Details</b> ).
t0	<b>Optional</b> dead time of the column, i.e. breakthrough time. Either given in <b>minutes</b> if t0_mode is "manual", or given as a single peak index included in which_peaks. Required for calculation of separation impedance $E$ (see <b>Details</b> ).
t0_mode	The mode to use for calculation of dead time. One of: "peak" or "manual", where $t0$ is taken directly as the time of specified peak maximum or given manually in minutes, respectively.
imped_met	A single character string specifying which method to use for calculation of Separation Impedance $E$ . One <b>or more</b> of: "all" (default), "indiv" and/or "univ". See <b>Details</b> .

## Details

The number of theoretical plates  $N$  is a widely-used column performance index where a higher number of theoretical plates represents better column efficiency. Several methods of calculation are implemented in **lcqc**. The full-width method uses the 'base width'  $W_b$  of a Gaussian peak defined as the baseline intercepts of tangent lines for a perfect Gaussian peak, which occur at **13.4% peak height**.

$$N_{FW} = 16(t_R/W_b)^2$$

The **5-sigma** method utilises the fact that a perfectly Gaussian peak is exactly 5-sigma (i.e. 5 standard deviations) wide at a 4.4% height (Bidlemeier & Warren Jr., 1984; Villalon, 2023) and uses the 5-sigma measure to determine width. This helps cope with tailing often encountered in chromatography, which distorts the peak from its Gaussian shape.

$$N_{5\sigma} = 25(t_R/W_{5\sigma})^2$$

The European Pharmacopoeia (EP) method is perhaps the most popular and uses the peak width at half-height ( $W_{50}$ ) along with retention time  $t_R$ . Since the coefficient used varies from 5.55 in the German (DAB), British (BP), and European (EP) Pharmacopoeias to 5.54 in the Japanese Pharmacopoeia (Rev. 15, April 2006), a mean of these (5.545) was implemented herein.

$$N_{EP} = 5.545(t_R/W_{50})^2$$

. #' An alternative equation uses the theoretical Gaussian inflection point width at ca. 60.7% peak height.

$$N_{inf} = 4(t_R/w_{60.7})^2$$

True to its name, the Area-Height (AH) method utilizes peak area  $A$  and height  $H$  to calculate  $N$ .

$$N_{AH} = 2\pi(t_R H/A)^2$$

All of the above methods are based on a true Gaussian peak shape, which is almost never encountered in practical chromatography. Tailing and (less prevalent) fronting phenomena distort the peak shape and cause errors in calculation. A simple equation that results in "approximately accurate" (Meyer, 2010) values of  $N$  and based on the Exponentially-Modified Gaussian (EMG) model was proposed by Foley & Dorsey (1983). The equation uses the peak width ( $W_{10}$ ) and half-widths at 10% height ( $b$  and  $a$  for trailing and leading width, respectively). The number of theoretical plates obtained via this equation is usually lower than that from other methods.

$$N_{EMG} = (41.7(t_R/W_{10})^2)/(b/a + 1.25)$$

Also calculated is the plate height, also known as Height Equivalent to a Theoretical Plate (HETP) is the distance in mm (or  $\mu\text{m}$ ) over which chromatographic equilibrium is achieved. This is simply related to  $N$  and column length  $L$  by:

$$HETP = L/N$$

From HETP, the reduced plate height  $h$  may also be calculated provided the average particle size of the stationary phase  $d_p$  (in  $\mu\text{m}$ ) is known;  $h$  is a dimensionless parameter and may be used to compare columns of different length and particle size more easily. Values of  $h$  should be in range of **2 to 5 (lower is better)**. For example, at a value of 3, complete chromatographic equilibrium is obtained over 3 layers of stationary phase (Meyer, 2010).

$$h = HETP/d_p$$

Finally, Separation Impedance  $E$  is a measure of column "quality" (efficiency) that incorporates back pressure ( $\Delta p$ ), number of theoretical plates, dynamic viscosity of the mobile phase ( $\eta$ , in mPas), and breakthrough time. Values of **<10000** are imperative for a liquid chromatography process to be considered "high performance" (Meyer, 2010). In **lcqc**,  $E$  may either be calculated separately for each individual value of  $N$  (much like  $HETP$  and  $h$ ) when `imped_met` includes "indiv". In this case, the following equation is used:

$$E = (\Delta p t_0)/(N^2 \eta)$$

Additionally, a universal equation independent of  $N$  may be used when `imped_met` includes "univ" (default).

$$E = (10^8/5.54^2) \times (\Delta p t_0/\eta) \times (W_{50}/t_R)^4$$

## Value

A named list of length 3 containing the data.frame of results for each examined peak (results), a character string of various information about the results (information), and the function call (call).

## References

- Bidlingmeyer, B.A., Warren, F.V.Jr. (1984), 'Column Efficiency Measurement', *Analytical Chemistry* **56** (14), pp. 1583A-1596A, DOI: <https://www.doi.org/10.1021/ac00278a002>.
- Foley, J.P., Dorsey, J.G. (1983), 'Equations for calculation of chromatographic figures of merit for ideal and skewed peaks', *Analytical Chemistry* **55** (4), pp. 730-737, DOI: <https://www.doi.org/10.1021/ac00255a033>.

Ishizuka, N., Kobayashi, H., Minakuchi, H., Nakahishi, K., Hirao, K., Hosoya, K., Ikegami, T., Tanaka, N. (2002), 'Monolithic silica columns for high-efficiency separations by high-performance liquid chromatography', *Journal of Chromatography A* **960**, pp. 85-96, DOI: [https://doi.org/10.1016/S0021-9673\(01\)01580-1](https://doi.org/10.1016/S0021-9673(01)01580-1).

Jarmalavičienė, R., Kornýšova, O., Westerlund, D., Maruška, A. (2003), 'Non-particulate (continuous bed or monolithic) restricted-access reversed-phase media for sample clean-up and separation by capillary-format liquid chromatography', *Analytical and Bioanalytical Chemistry* **377** (5), pp. 902-908, DOI: <https://www.doi.org/10.1007/s00216-003-2244-z>.

Meyer, V.R. (2010), *Practical High-Performance Liquid Chromatography*, John Wiley & Sons, Chichester, United Kingdom.

The United States Pharmacopeial Convention (2021), *Physical Tests/Chromatography* (USP 40-NF 35), available at: <https://www.usp.org/harmonization-standards/pdg/excipients/chromatography> (accessed 24.04.2024).

Villalon, G.C. (2023), 'Characterization of Chromatographic Peaks with Excel', *Journal of Chemical Education* **100**, pp. 928-932, DOI: <https://doi.org/10.1021/acs.jchemed.2c00588>.

## See Also

[chrom\\_detect](#), [chrom\\_visc](#)

## Examples

```
## Not run:
#Get main data and peak areas
dt <- lcqc::wf_detpeaks
pas <- lcqc::wf_ints$integ_res$pa

#Theoretical plates only
res <- chrom_tplate(dt, method = c("S5", "EP", "EMG"))

#With AH method, HETP, and h
res2 <- chrom_tplate(dt, method = "AH", pa = pas, len = 150, dp = 5)

#With Separation Impedance E
res3 <- chrom_tplate(dt, method = "AH", pa = pas, len = 150, dp = 5, visc = 0.5264825,
t0 = 0.25, t0_mode = "manual", deltap = 85, imped_met = "all")

## End(Not run)
```

---

chrom\_width

*Auto-estimate smoothing width and number of smoothing passes*

---

## Description

Part of the [chrom\\_detect](#) workflow. Estimates smoothing window width, number of smoothing passes, average and minimum peak width at 5% height, minimum width at inflection points.

**Usage**

```
chrom_width(
  rtime,
  sig,
  start_smooth = c(7, 1),
  bcorr = "none",
  bpars = "default",
  ampfrac = 1,
  smooth_method = rep("tri", 2)
)
```

**Arguments**

rtime	A numeric vector of retention time (x-axis) values.
sig	A numeric vector of signal (y-axis) values.
start_smooth	A numeric vector of length 2 containing, in that order, the <b>odd</b> number of smoothing points and passes to use for <b>initial smoothing</b> . Defaults to c(7, 1).
bcorr	Method to use for baseline correction (defaults to "none"), see <a href="#">chrom_bline</a> for available options.
bpars	Parameters to use for baseline correction when bcorr!="none".
ampfrac	The numeric quantile percentage value (from 0.001 to 50) to use for amplitude limit estimation via function <a href="#">chrom_amplim</a> .
smooth_method	A character vector of length 2 providing smoothing methods for signal and derivatives (in that order). If a vector of length 1 is given, the same method is applied to both types of data. See <a href="#">chrom_smooth</a> for available methods.

**Value**

A named numeric vector of length 5 containing the following elements:

- "points"** Number of auto-estimated smoothing points (smooth width).
- "passes"** Number of auto-estimated smoothing passes.
- "width\_5\_mean"** **Average** peak width at 5% of peak height (in number of points/indices).
- "width\_5\_min"** **Minimum** peak width at 5% of peak height (in number of points/indices).
- "width\_inf"** Minimum inflection point width (in number of points/indices).

**See Also**

[chrom\\_detect](#), [acc\\_inf](#)

**Examples**

```
dt <- lcqc::simlc1
chrom_width(dt[, "Time"], dt[, "Signal"], ampfrac = 0.05)
```



---

cprint	<i>Print information about function run</i>
--------	---

---

### Description

Prints the contents of `$information` contained in the output of various **lcgc** functions.

### Usage

```
cprint(x)
```

### Arguments

x	Output from one of: <code>chrom_detect</code> , <code>chrom_skim</code> , <code>chrom_icf</code> , <code>chrom_visc</code> , <code>chrom_tplate</code> , <code>chrom_asym</code> , <code>chrom_retf</code> , <code>chrom_sepf</code> , <code>chrom_res</code> , or <code>chrom_addmets</code> .
---	---

### Value

A cat printout in the console.

### Examples

```
## Not run:  
cprint(lcgc::wf_detpeaks)  
  
## End(Not run)
```

---

dtprep	<i>Summarize and prepare peak detection data for integration</i>
--------	--

---

### Description

This function takes peak detection results output of `chrom_detect` and prepares the data for integration via functions `chrom_skim` and/or `chrom_icf`.

### Usage

```
dtprep(input, vars = "auto", crit_w = "auto")
```

### Arguments

input	The output of function <code>chrom_detect</code> .
vars	Either "auto" (default) or a character vector of length 3 specifying the column names of retention time, signal, and second derivative data in input (in that order).
crit_w	Critical width parameter to use for baseline construction via <code>fastchrom_bline</code> .

Value

A list of length 5 containing the following elements:

- main\_df** A data.frame containing data indices, retention time, original and baseline-corrected signal (output from `fastchrom_bline`).
- type\_df** A list of 5 data.frame objects summarizing types, original, baseline-corrected (`fastchrom_bline`), and/or accurately-interpolated (`acc_inf`, `acc_max`) values of peak starts, ends, linfo, rinfo, maxes.
- grp\_df** Data equivalent to that in `type_df` but organized by groups of baseline-resolved peaks in a nested list.
- grp\_blines** A data.frame containing **baseline-resolved peak group IDs**, original xy-coordinates, FastChrom baselines, baseline-corrected signal, and second derivatives of all peak regions.
- acc\_tops** A data.frame containing combined results of accurate inflection point and peak retention time determination via `acc_inf` and `acc_max`, respectively.
- peak\_list** A list of data.frame objects, each containing retention time, original signal, baselines, baseline-corrected signal, second derivative, and peak ID data for an individual peak.

See Also

`chrom_skim`, `chrom_icf`, `chrom_detect`, `acc_inf`, `acc_max`

Examples

```
## Not run:
dtprep(lcqc:::wf_detpeaks)

## End(Not run)
```

exgc1	<i>Experimental GC-MS Chromatogram (example 1)</i>
-------	--

Description

An experimental example two column GC-MS chromatogram (1 of 2)

Usage

`exgc1`

Format

**exgc1:**  
A data frame with 2141 rows and 2 columns  
**Time** Retention time of chromatogram  
**Signal** Signal of chromatogram

---

exgc2*Experimental GC-MS Chromatogram (example 2)*

---

**Description**

An experimental example two column GC-MS chromatogram (2 of 2)

**Usage**

exgc2

**Format**

exgc2:

A data frame with 2141 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

exlc1*Experimental LC Chromatogram (example 1)*

---

**Description**

An experimental example two column HPLC chromatogram (1 of 11)

**Usage**

exlc1

**Format**

exlc1:

A data frame with 1301 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

exlc10

*Experimental LC Chromatogram (example 10)*

---

### Description

An experimental example two column HPLC chromatogram (10 of 11)

### Usage

exlc10

### Format

exlc10:

A data frame with 1102 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

exlc11

*Experimental LC Chromatogram (example 11)*

---

### Description

An experimental example two column HPLC chromatogram (11 of 11)

### Usage

exlc11

### Format

exlc11:

A data frame with 3282 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

exlc2*Experimental LC Chromatogram (example 2)*

---

**Description**

An experimental example two column HPLC chromatogram (2 of 11)

**Usage**

exlc2

**Format**

exlc2:

A data frame with 1301 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

exlc3*Experimental LC Chromatogram (example 3)*

---

**Description**

An experimental example two column HPLC chromatogram (3 of 11)

**Usage**

exlc3

**Format**

exlc3:

A data frame with 1301 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

exlc4

*Experimental LC Chromatogram (example 4)*

---

### Description

An experimental example two column HPLC chromatogram (4 of 11)

### Usage

exlc4

### Format

exlc4:

A data frame with 1301 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

exlc5

*Experimental LC Chromatogram (example 5)*

---

### Description

An experimental example two column HPLC chromatogram (5 of 11)

### Usage

exlc5

### Format

exlc5:

A data frame with 1353 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

exlc6*Experimental LC Chromatogram (example 6)*

---

**Description**

An experimental example two column HPLC chromatogram (6 of 11)

**Usage**

exlc6

**Format**

exlc6:

A data frame with 4701 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

exlc7*Experimental LC Chromatogram (example 7)*

---

**Description**

An experimental example two column HPLC chromatogram (7 of 11)

**Usage**

exlc7

**Format**

exlc7:

A data frame with 664 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

exlc8*Experimental LC Chromatogram (example 8)*

---

**Description**

An experimental example two column HPLC chromatogram (8 of 11)

**Usage**

exlc8

**Format**

exlc8:

A data frame with 1596 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

exlc9*Experimental LC Chromatogram (example 9)*

---

**Description**

An experimental example two column HPLC chromatogram (9 of 11)

**Usage**

exlc9

**Format**

exlc9:

A data frame with 1436 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram



---

ext_lcqc	<i>Browse external package data.</i>
----------	--------------------------------------

---

**Description**

Lists paths or filenames of external raw data files (such as .csv and .txt) included with the **lcqc** package.

**Usage**

```
ext_lcqc(full.path = TRUE, txt_only = TRUE)
```

**Arguments**

full.path	A logical switch. Should full paths to external files be returned? Defaults to TRUE.
txt_only	A logical switch.. Should only <b>.txt</b> files be listed? When FALSE, all external files included with the package are listed.

**Value**

Filenames (or file paths if full.path = TRUE) of external data files included with the package.

**Examples**

```
ext_lcqc(full.path = FALSE)
```

---

fastchrom_bline	<i>Calculate baselines of resolved peaks using the FastChrom algorithm</i>
-----------------	--

---

**Description**

The function calculates baselines of well-resolved peaks and peak groups via the FastChrom algorithm of Johnsen et al. (2013).

**Usage**

```
fastchrom_bline(inds = NA, sig, starts, ends, crit_w = 1, for_plot = FALSE)
```

**Arguments**

inds	An <b>optional</b> numeric vector of data indices equal in length to sig. Equal to seq(length(sig)) by default.
sig	Chromatographic signal data as a numeric vector.
starts, ends	A numeric vector of peak <b>start</b> and <b>end</b> indices.
crit_w	The critical width parameter. Usually equal to the minimum peak width at half height (see <b>Details</b> ).
for_plot	A logical toggle that determines whether the output data should be formatted for plotting (e.g. via <a href="#">chrom_plot</a> ).

## Details

Iteratively derives baselines for all resolved regions based on the FastChrom algorithm proposed by Johnsen et al. (2013). First, all peak regions (e.g. determined by `chrom_detect`) are separated and a linear interpolated is carried out between their starting and ending points to derive initial baselines. Each baseline is then checked and, if necessary, adjusted iteratively as follows: if a **consecutive** number of baseline points equal to or greater than the critical width (`crit_w`) of the baseline are **above the signal**, the baseline is forced through the **single lowest** point relative to the baseline and re-interpolated. This is repeated until there are no baseline points above the signal.

## Value

The output changes depending on whether `for_plot` is TRUE. If so, a list of length 3 is returned containing numeric vectors of the Original\_Signal, Corrected\_Signal, and Baseline. Otherwise, the same data alongside data indices is returned as a 4-column data.frame.

## References

Johnsen, L.G., Skov, T., Houlberg, U., Bro, R. (2013), 'An automated method for baseline correction, peak finding and peak grouping in chromatographic data', *Analyst* **138** (12), pp. 3502-3511, DOI: <https://www.doi.org/10.1039/C3AN36276K>

## See Also

`dtprep`

## Examples

```
sigvec <- lcqc::simlc1["Signal"]
strvec <- c(486, 763, 916, 1745, 2428)
endvec <- c(707, 897, 1050, 1946, 2588)
crw <- 10

#Data formatted simply
res1 <- fastchrom_bline(sig = sigvec, starts = strvec, ends = endvec, crit_w = crw,
for_plot = FALSE)

#Data formatted for plotting
res2 <- fastchrom_bline(sig = sigvec, starts = strvec, ends = endvec, crit_w = crw, for_plot = TRUE)

#Create plot
plot(res2[["Original_Signal"]])
lines(res2[["Baseline"]], col = "red")
```

---

find\_peaks

*Detect maxima in vector*

---

## Description

Detects maximum values (peaks) in a series of numbers.

**Usage**

```
find_peaks(x, m = 3)
```

**Arguments**

**x** Numeric vector in which to detect maxima.

**m** The number of data points on either side (i.e. neighbourhood of points) used to validate maxima and minima.

**Details**

The function may also be used to detect minima by simple passing the inverse of a vector, i.e.  $-x$ .

**Value**

Position(s) of maxima in vector.

**Examples**

```
vec <- c(1,2,3,2,1,1,2,1)
find_peaks(vec, m = 3)
find_peaks(-vec, m = 3)
```

---

icf\_plot

*Visualize the results of Non-Linear Least Squares Iterative Curve Fitting (ICF)*

---

**Description**

Visualizes the results of Non-Linear Least Squares Iterative Curve Fitting (ICF) of chromatographic data obtained via [chrom\\_icf](#).

**Usage**

```
icf_plot(
  input,
  cols = "default",
  plabs = "default",
  plotsum = TRUE,
  plotind = FALSE,
  fillplot = TRUE,
  txtmax = TRUE,
  norm = FALSE,
  asprat = 0.71
)
```

**Arguments**

input	The data.frame of data output from the <code>[[ "main_data" ]]</code> element of <code>chromskim</code> .
cols	Plot colour palette. Either "default" or a <b>named</b> character vector of colours. Names denote the following plot elements: original chromatogram ("main"), Gaussian ("gs"), EGH ("egh"), EMG ("emg"), and ETG ("etg") grouped models, unmodeled peaks ("none"), and models for individual peaks rather than groups ("indiv").
plabs	Plot labels. Either "default" or a <b>named</b> character vector of labels for the following elements: main plot title ("main"), retention time ("x"), and signal ("y").
plotsum	A logical switch. Should summed models for peak groups be plotted? Defaults to TRUE.
plotind	A logical switch. Should individual peak models be plotted? Defaults to FALSE.
fillplot	A logical specifying whether peak models should be coloured/filled (TRUE by default). When FALSE, only outlines of peak models are plotted.
txtmax	A logical switch specifying whether to plot peak numbers at apices. Defaults to FALSE. Only applies to plots of individual peak models (plotind = TRUE).
norm	A logical specifying whether to normalize the chromatographic signal (y-axis) to a percentage scale (0 to 100). Defaults to FALSE.
asprat	Aspect ratio of the plot (defaults to 0.71).

**Value**

A ggplot-class object containing the plot.

**See Also**

`chrom_icf`

**Examples**

```
## Not run:
#Get data and plot
icf_res <- lcqc::wf_icf[["main_data"]]
icf_plot(icf_res)

#Normalized plot with models for individual peaks included
icf_plot(icf_res, plotind = TRUE, norm = TRUE)

## End(Not run)
```

---

insect	<i>Find intersection between two lines</i>
--------	--

---

**Description**

Locates the intersection point of two lines characterized by a slope and intercept.

**Usage**

```
insect(l1, l2)
```

**Arguments**

l1	Line 1 characterized by slope and intercept. A numeric vector of length 2 in the format: c(intercept, slope).
l2	Line 2 characterized by slope and intercept. Format is identical to that of l1.

**Value**

A numeric vector with c(x,y) coordinates of the intersection point (if any).

**See Also**

[slope](#), [intercept](#)

**Examples**

```
#Define two lines by slope and intercept
#Format: c(intercept, slope)
l1 <- c(10, -2)
l2 <- c(0, 2)
isect <- insect(l1, l2)
```

---

integ_plot	<i>Visualize traditional integration results</i>
------------	--

---

**Description**

Visualizes the results of traditional integration of data obtained from [chrom\\_skim](#), including the peak apices, group baselines, as well as perpendicular drop and/or tangent/exponential/Gaussian front or tail skim lines.

**Usage**

```
integ_plot(
  input,
  cols = "default",
  plabs = "default",
  norm = FALSE,
  plot_max = FALSE,
  txt_max = FALSE,
  asprat = 0.71
)
```

Arguments

input	The list of data output from <a href="#">chrom_skim</a> .
cols	Plot colour palette. Either "default" or a <b>named</b> character vector of colours. Names denote the following plot elements: main chromatogram ("main"), peak group baselines ("grp"), perpendicular droplines ("pd"), tangent skim lines ("ts"), exponential skim lines ("es"), and Gaussian skim lines ("gs").
plabs	Plot labels. Either "default" or a <b>named</b> character vector of labels for the following elements: main plot title ("main"), retention time ("x"), and signal ("y").
norm	A logical specifying whether to normalize the chromatographic signal (y-axis) to a percentage scale (0 to 100). Defaults to FALSE.
plot_max	A logical switch specifying whether to show peak apices with shape markers. Defaults to FALSE.
txt_max	A logical switch specifying whether to plot peak numbers at apices. Defaults to FALSE.
asprat	Aspect ratio of the plot (defaults to 0.71).

Value

A ggplot-class object containing the plot.

See Also

[chrom\\_skim](#)

Examples

```
## Not run:
#Get and plot data
skim_res <- lcqc::wf_ints
integ_plot(skim_res, plot_max = TRUE)

#With y-axis normalisation to 0-100 and text peak labels
integ_plot(skim_res, plot_max = TRUE, plabs = c(main = ""), txt_max = TRUE, norm = TRUE)

## End(Not run)
```

---

is.even	<i>Find whether a number is even or odd</i>
---------	---

---

Description

Returns a logical values based on whether an input number is even or odd.

Usage

```
is.even(x)

is.odd(x)
```

**Arguments**

`x`                      Single numeric integer value.

**Value**

A TRUE/FALSE depending on whether the input is an odd (for `is.odd`) or even (for `is.even`) number.

**Examples**

```
is.even(1)
is.odd(3)
```

---

linspline

*Find roots via linear or non-linear interpolation*


---

**Description**

By default, function `linspline` finds the zero-crossing (i.e. root) of a linear interpolation function. Similarly, `cubspline` finds the root of a cubic spline (output from `stats::splinefun`).

**Usage**

```
linspline(x, y, y0 = 0, verbose = TRUE)

cubspline(f, y0 = 0, verbose = TRUE)
```

**Arguments**

`x`                      The known x values (only for `linspline`).  
`y`                      The known y values (only for `linspline`).  
`y0`                     The y value for which an x value is to be found (defaults to 0, indicating the root is to be found).  
`verbose`              A logical which determines whether the results are plotted using `plot()`.  
`f`                      Cubic spline results obtained from `stats::splinefun`.

**Value**

A numeric vector of intersection point(s) along the x-axis.

**Author(s)**

Zheyuan Li (see also [here](#))

**Examples**

```
set.seed(0)
x <- 1:10 + runif(10, -0.1, 0.1)
y <- rnorm(10, 3, 1)
f <- stats::splinefun(x, y, method = "fmm")
linspline(x, y, y0 = 2.85, verbose = FALSE)
cubspline(f, y0 = 2.85, verbose = FALSE)
```

noise\_plot

*Visualise noise data***Description**

Create a summary plot of noise data, usually derived from [chrom\\_derlims](#).

**Usage**

```
noise_plot(
  noise,
  thres,
  plot_title = "Peak Recognition Thresholds",
  y_lab = "Noise",
  asprat = 0.71
)
```

**Arguments**

noise	A numeric vector containing noise data.
thres	A numeric vector of length 2 containing the lower and upper noise thresholds.
plot_title	The character plot title.
y_lab	The character y-axis title.
asprat	Aspect ratio of the plot.

**Value**

A ggplot-class plot object.

**See Also**

[chrom\\_derlims](#)

**Examples**

```
#Get noise and threshold data
## Not run:
dt <- lcqc::wf_detpeaks[["results"]]
nsdt <- dt[["Derivative_Noise"]][["SD_noise"]]
thrdt <- dt[["Derivative_Limits"]][["SD"]]
noise_plot(nsdt, thrdt)

## End(Not run)
```



---

read_shim	<i>Read Shimadzu LabSolutions GC-FID, GC-MS, and HPLC data from ASCII file</i>
-----------	--

---

## Description

Reads ASCII files originating from Shimadzu LabSolutions and GCMS Solutions software and extracts the chromatogram, peak table, metadata, and MS similarity search table. Modified and extended from function `read_shimadzu` from the **chromConverter** package.

## Usage

```
read_shim(
  file,
  ptable = TRUE,
  simtable = TRUE,
  pnames = TRUE,
  pcas = TRUE,
  metadata = TRUE,
  mode = "fid",
  sep = "auto",
  decsep = "auto",
  fix_names = TRUE,
  fil_cols = TRUE,
  cols = "default",
  chromcols = "default",
  rm_dups = FALSE,
  unify_cols = NA,
  trange = NA
)
```

## Arguments

file	The character file path to a compatible Shimadzu ASCII.
ptable	A logical specifying whether the peak table is extracted. Defaults to TRUE.
simtable	A logical specifying whether the GC-MS similarity table is extracted. Only valid when mode = "gcms". Defaults to TRUE.
pnames	A logical specifying whether peak names are retrieved (where available). TRUE by default.
pcas	A logical specifying whether CAS numbers corresponding to peak names are retrieved. Only valid when mode = "gcms". TRUE by default.
metadata	A logical specifying whether metadata is retrieved (defaults to TRUE).
mode	A character value of the type of chromatogram to be read. One of: "fid" (GC-FID; default), "gcms" (GC-MS), or "lc" (HPLC).
sep, decsep	A single character indicating the <b>column</b> and <b>decimal</b> ("decsep") separator of input ASCII files. The default value "auto" attempts to determine this automatically.
fix_names	A logical specifying whether column names should be made syntactically correct (TRUE by default).

fil_cols	A logical specifying whether to truncate the peak table to only include the most relevant columns (TRUE by default).
cols, chromcols	Either "default" or a character vector of column names to extract from the <b>peak table</b> (cols) or <b>chromatogram</b> (chromcols).
rm_dups	A logical specifying whether duplicates should be removed from the GC-MS peak table. Only valid when mode = "gcms". Defaults to FALSE by default.
unify_cols	Either NA (default) or an <b>ordered</b> character vector of <b>length 8</b> containing universal column names to assign for all function output and corresponding to the following variables: peak number, retention time, start time, end time, area, height, area/height ratio, and compound name.
trange	Either NA (default) or a numeric vector of length <b>1</b> or <b>2</b> specifying the <b>lower</b> and/or <b>upper</b> retention time limit(s) used to truncate the chromatogram and peak table.

### Value

A named list containing various data extracted from the ASCII file. These include metadata, chromatogram (one or more), ptable (peak table), simtable (MS similarity table), pnames (a character vector of peak names), and pcas (a character vector of corresponding CAS numbers). When mode = "gcms" and Selected Ion Monitoring (SIM) chromatograms are present, they are all extracted alongside the Total Ion Current (TIC) chromatogram as separate list elements. Similarly, all wavelength-specific LC chromatograms are extracted separately when mode = "lc".

### References

Bass, E. (2022), *chromConverter*, available at: <https://cran.r-project.org/package=chromConverter> (accessed 25.04.2024).

### See Also

[ext\\_lcqc](#)

### Examples

```
## Not run:
#Get data paths of external Shimadzu ASCII data included with lcqc
dtp <- ext_lcqc()

#Read FID chromatogram
fidc <- read_shim(dtp[grepl("exgc_",dtp)[1]], mode = "fid")

#Read GCMS chromatogram and peak table
msc <- read_shim(dtp[grepl("exgcms_",dtp)[1]], mode = "gcms")

#Read HPLC chromatogram
lcc <- read_shim(dtp[grepl("exlc_",dtp)[1]], mode = "lc")

## End(Not run)
```

render\_defs

*Render LCQC data into a concise HPLC column performance report*

## Description

Compiles key data obtained from the **lcqc** workflow into a modular HPLC Column Performance Report in .PDF format. The function utilizes the **Quarto** framework for R.

## Usage

```
render_defs()

lcqc_render(
  tpl,
  cplot,
  tpars,
  spec,
  clogo = NA,
  sym = NA,
  ext = NA,
  mets = c("EP", "AH", "Tf", "As"),
  exmets = c("Linear Velocity", "Packing Porosity", "Flow Resistance", "Permeability"),
  add_tpa = FALSE,
  expath = getwd(),
  asprat = 0.4,
  pnms = NA,
  pconcs = NA,
  add_pnums = TRUE,
  fontsize = 10,
  which_pks = "all"
)
```

## Arguments

- |       |   |
|-------|---|
| tpl   | The output of function <a href="#">chrom_tplate</a> containing theoretical plates and other metrics.  |
| cplot | A ggplot of the chromatogram to be included in the report. Usually obtained via the <a href="#">chrom_skim</a> or <a href="#">chrom_icf</a> workflows.  |
| tpars | A <b>named</b> character vector specifying various general text to be included in the report. Values may include (also run <a href="#">render_defs</a> for examples): <ol style="list-style-type: none"> <li>1. Column serial number "sn"</li> <li>2. Column part number "pn"</li> <li>3. Short column description "desc"</li> <li>4. Batch number "bn"</li> <li>5. Document number "dnum"</li> <li>6. Operator name and surname "oper"</li> <li>7. Mobile phase specification "mp"</li> <li>8. Shipment phase "sp" (which solvent is the column shipped in?)</li> <li>9. Back pressure "bp"</li> </ol> |

	10. Flow rate "flow"
	11. Column temperature "temp"
	12. Injection volume "inj"
	13. Analyte concentration unit "unit_c"
	14. Timestamp for the report genesis "tform" ("auto" by default)
spec	A <b>named</b> list of QC specifications to display in the report. Names <b>must match</b> the elements of mets.
clogo	A character filepath to a custom <b>.JPG</b> logo to include in the report header. A placeholder is displayed if no logo is provided.
sym	<b>Optional</b> asymmetry metrics and Total Peak Analysis output from function <a href="#">chrom_asym</a> .
ext	<b>Optional</b> additional column-specific metrics output from function <a href="#">chrom_addmets</a> .
mets	A character vector of theoretical plate ( $N$ ) and asymmetry metrics to display in the report. Possible metrics are: European Pharmacopoeia $N$ ("EP"), Area-Height $N$ ("AH"), sigma-5 $N$ ("S5"), Exponentially-Modified Gaussian $N$ ("EMG"), Asymmetry Factor ("As"), and/or USP/EP Tailing Factor ("Tf").
exmets	<b>Optional</b> character vector of <b>additional</b> column-specific metrics to display at the bottom of the first page. <b>Only relevant if</b> ext is provided. Possible values are "Linear Velocity", "Packing Porosity", "Flow Resistance", and/or "Permeability".
add_tpa	A logical switch indicating whether to add TPA plots to the second page of the report (defaults to FALSE).
expath	A character directory path into which to export the .PDF report.
asprat	Aspect ratio of the chromatogram plot (0.4 by default).
pnms	Either NA (default) or a character vector of peak names to include in the report. Must be equal in length to the total number of peaks in the input data, or to which_pks if these are specified.
pconcs	Either NA (default) or a vector of concentrations corresponding to each peak included in the data, or to those specified in which_pks.
add_pnums	A logical switch specifying whether to add peak index numbers to the PDF report <b>when pnms are given</b> . Defaults to TRUE.
fontsize	The font size to use for main text of the report (10 by default).
which_pks	Either "all" (default) or a numeric vector of peak indices to include in the report.

### Value

The column performance report .PDF file is exported to the directory specified in expath.

### References

Allaire, J.J., Teague, C., Scheidegger, C., Xie, Y., Dervieux, C., (2024), *Quarto*, version 1.4, DOI: <https://www.doi.org/10.5281/zenodo.5960048>.

### See Also

[render\\_defs](#), [chrom\\_tplate](#), [chrom\\_asym](#), [chrom\\_addmets](#), [chrom\\_skim](#)

## Examples

```
## Not run:
#View available definitions
render_defs()

#Render .PDF report into the working directory
lcqc_render(tpl = lcqc::wf_tplate,
cplot = lcqc::wf_ints[["plot"]], #ALWAYS remove title and legend from plot
sym = lcqc::wf_asyms,
ext = lcqc::wf_perfmets,
which_pks = 4:7,
pnms = c("Naphthalene", "Anthracene", "Caffeine", "Benzene"),
pconcs = c(250, 500, 250, 500),
mets = c("EP", "AH", "Tf", "As"),
exmets = c("Linear Velocity", "Packing Porosity", "Flow Resistance", "Permeability"),
add_tpa = TRUE,
spec = list(EP = rep(">4000", 4),
            AH = rep(">4000", 4),
            Tf = rep("0.8-1.2", 4),
            As = rep("0.8-1.2", 4)),
tpars = c(sn = "C18-2024000001",
pn = "L-C182546510",
desc = "AC-C18 Column\n(250x4.6 mm, 5 micron, 10 nm)",
bn = "2024000001",
dnum = "LC-C18-2024000001",
oper = "Deniz Can Koseoglu",
mp = "70:30 MeOH:Water",
sp = "80:20 (volumetric) IPA:Water",
bp = "126 bar",
flow = "1.0 mL/min",
temp = "30 degC",
inj = "20 microL",
unit_c = "ppm",
tform = "default"),
expath = paste0("C:/Users/Denizcan/Desktop"),
asprat = 0.4,
add_pnums = TRUE,
fontsize = 10)

## End(Not run)
```

---

sdmin\_infchk

*Reject those SD minima without proximal inflection points*


---

## Description

**This function is not exported.** Part of the [chrom\\_detect](#) workflow, not intended for standalone use.

Removes SD minima which do not have any proximal inflection points, prior to peak detection.

## Usage

```
sdmin_infchk(sdmin, sigmax, input)
```

**Arguments**

sadmin	A numeric vector of SD minima indices.
sigmax	A numeric vector of signal maxima indices.
input	The output of <a href="#">markmerge</a> containing all peak markers.

**Value**

A numeric vector of filtered sadmin values.

**See Also**

[chrom\\_detect](#), [peakfind](#)

---

simlc1	<i>Simulated LC Chromatogram (example 1)</i>
--------	--

---

**Description**

A simulated example two column HPLC chromatogram (1 of 7)

**Usage**

```
simlc1
```

**Format**

```
simlc1:
```

A data frame with 3000 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

simlc2	<i>Simulated LC Chromatogram (example 2)</i>
--------	--

---

**Description**

A simulated example two column HPLC chromatogram (2 of 7)

**Usage**

```
simlc2
```

**Format**

```
simlc2:
```

A data frame with 3000 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

`simlc3`*Simulated LC Chromatogram (example 3)*

---

**Description**

A simulated example two column HPLC chromatogram (3 of 7)

**Usage**`simlc3`**Format**`simlc3:`

A data frame with 3000 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

`simlc4`*Simulated LC Chromatogram (example 4)*

---

**Description**

A simulated example two column HPLC chromatogram (4 of 7)

**Usage**`simlc4`**Format**`simlc4:`

A data frame with 3000 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

`simlc5`*Simulated LC Chromatogram (example 5)*

---

**Description**

A simulated example two column HPLC chromatogram (5 of 7)

**Usage**`simlc5`**Format**`simlc5:`

A data frame with 3000 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

`simlc6`*Simulated LC Chromatogram (example 6)*

---

**Description**

A simulated example two column HPLC chromatogram (6 of 7)

**Usage**`simlc6`**Format**`simlc6:`

A data frame with 3000 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram



---

simlc7	<i>Simulated LC Chromatogram (example 7)</i>
--------	--

---

**Description**

A simulated example two column HPLC chromatogram (7 of 7)

**Usage**

```
simlc7
```

**Format**

**simlc7:**

A data frame with 3000 rows and 2 columns

**Time** Retention time of chromatogram

**Signal** Signal of chromatogram

---

slope	<i>Slope and intercept of a linear function</i>
-------	---

---

**Description**

Functions to calculate the slope and intercept of a line.

**Usage**

```
slope(x, y)
```

```
intercept(x, y)
```

**Arguments**

**x** Numeric vector of known x values.

**y** Numeric vector of Known y values.

**Value**

A numeric value of the linear gradient/slope.

**See Also**

<https://stackoverflow.com/questions/66771929/intercept-and-slope-functions-in-r>

**Examples**

```
with(CO2, slope(conc, uptake))
with(CO2, intercept(conc, uptake))
```

---

smooth_whit	<i>Weighted (Whittaker-Henderson) smoother</i>
-------------	--

---

### Description

Smooth a signal with a finite difference penalty of order 2. For more details, see **References**. Modified from package **ptw**.

### Usage

```
smooth_whit(input, lambda, wts = rep(1, length(input)))
```

### Arguments

input	Input numeric vector containing signal for smoothing.
lambda	Smoothing parameter (defaults to 1600). Degree of smoothing increases with value.
wts	Vector of relative weights, must equal input in length. Defaults to 1.

### Value

Numeric vector containing the smoother signal.

### References

Eilers, P.H.C. (2003), 'A perfect smoother', *Analytical Chemistry* **75**, pp. 3631-3636.  
 Eilers, P.H.C. (2004), 'Parametric Time Warping', *Analytical Chemistry* **76** (2), pp. 404-411.

### See Also

[bline\\_als](#)

### Examples

```
smooth_whit(lcqc::simlc1["Signal"], lambda = 1600)
```

---

supp_pars	<i>Supplement function arguments (named vectors) with default values</i>
-----------	--

---

### Description

Compares an input **named** vector of parameters with a default vector, supplementing with default parameters where necessary. **This function is not exported.**

### Usage

```
supp_pars(defpars, pars = "default", parlb = NA)
```

**Arguments**

defpars	A <b>named</b> vector of default parameters (numeric or character).
pars	A <b>named</b> vector of input parameters where all names must correspond to those of defpars.
parlb	The label for parameters. If NA (default), defaults to "pars".

**Value**

The input vector supplemented with default parameters (if necessary).

---

z_optim	<i>Optimization of z-score thresholding algorithm</i>
---------	---

---

**Description**

The function tests a specific range of input values for the **lcqc** z-scores peak picking algorithm (see [z\\_thres](#)) using a numeric vector. The combination of parameters resulting in the highest number of legitimate peaks (along a numeric vector x) at the lowest values of lag and threshold is selected. These parameters are returned alongside the start and end indices of corresponding detected peak regions.

**Usage**

```
z_optim(
  x,
  lag_rng = seq(0.01, 0.05, 0.005),
  thres_rng = seq(1, 6, 0.5),
  peak_thres = 0.02,
  noise_thres = 0.005,
  mode = "percent"
)
```

**Arguments**

x	Input numeric vector of equi-spaced signal values.
lag_rng	Sequence of lag values to test, as outlined in <a href="#">z_thres</a> .
thres_rng	Sequence of threshold values to test, as outlined in <a href="#">z_thres</a> .
peak_thres	Threshold value. Minimum number (or fraction) of consecutive points in x required to initially accept a peak.
noise_thres	Threshold value. Minimum number (or fraction) of consecutive points in x required to keep a peak in the final stage of optimization. Non-conforming peaks (which are too narrow) are removed as noise.
mode	One of "percent" or "absolute". Determines whether peak_thres and noise_thres are interpreted as fractions of total points in x, or as absolute values.

**Value**

A list of length two containing the following elements:

**Peak\_Limits** A 2-column data.frame containing the start and end indices of detected peaks along x.

**Best\_ZScore\_Run** The results of the "best" z-score thresholding run, formatted as per the [z\\_thres](#) function.

**See Also**

[z\\_thres](#)

**Examples**

```
yvals <- lcqc::simlc1[, "Signal"]
z_optim(yvals)
```

---

z_thres	<i>Thresholding based on z-scores</i>
---------	---------------------------------------

---

**Description**

Simple, robust peak picking algorithm based on calculating and thresholding z-scores.

**Usage**

```
z_thres(y, lag, threshold, influence)
```

**Arguments**

y	Numeric vector of equi-spaced y-values. Length needs to be $\geq \text{lag} + 2$ .
lag	Number of observations to base average and standard deviation on (e.g. 5).
threshold	The z-score (factor to multiply standard deviation of the moving average by) at which the algorithm signals the presence of positive or negative peaks (e.g. 3.5).
influence	Factor between 0 and 1 denoting the relative influence/weight that new data points have on the calculation of moving average and moving standard deviation (e.g. 0.5).

**Value**

A list of 4 elements:

**signals** Vector of length equal to y denoting presence or absence of peaks with three possible values: 0 (no peak), 1 (positive peak), -1 (negative peak).

**avgFilter** Vector of length equal to y containing moving averages.

**stdFilter** Vector of length equal to y containing moving standard deviations.

**params** A vector of length 3 containing the used lag, threshold, and influence parameters passed to the function.

## References

Brakel, J.P.G. van (2014). "Robust peak detection algorithm using z-scores". Stack Overflow. Available at: <https://stackoverflow.com/questions/22583391/peak-signal-detection-in-realtime-timeseries-22640362#22640362> (version: 2020-11-08).

## See Also

[z\\_optim](#)

## Examples

```
z_thres(lcqc::exlc1[, "Signal"], lag = 13, threshold = 4, influence = 0)
```

# Index

## \* datasets

- exgc1, [50](#)
- exgc2, [51](#)
- exlc1, [51](#)
- exlc10, [52](#)
- exlc11, [52](#)
- exlc2, [53](#)
- exlc3, [53](#)
- exlc4, [54](#)
- exlc5, [54](#)
- exlc6, [55](#)
- exlc7, [55](#)
- exlc8, [56](#)
- exlc9, [56](#)
- simlc1, [70](#)
- simlc2, [70](#)
- simlc3, [71](#)
- simlc4, [71](#)
- simlc5, [72](#)
- simlc6, [72](#)
- simlc7, [73](#)
  
- acc\_inf, [3](#), [5](#), [48](#), [50](#)
- acc\_max, [4](#), [4](#), [40](#), [50](#)
- addnms, [5](#), [24](#)
- apexbnds, [24](#)
  
- bline\_als, [6](#), [9](#), [19](#), [20](#), [74](#)
- bline\_chang, [7](#), [9](#), [19](#), [20](#)
- bline\_isrea, [8](#), [9](#), [19](#), [20](#)
- bline\_plot, [9](#), [19](#), [20](#)
- bline\_poly, [9](#), [10](#), [19](#), [20](#)
- browse\_visc, [11](#), [11](#)
  
- chk\_pack, [13](#)
- chkdt, [41](#)
- chrom\_addmets, [14](#), [27](#), [68](#)
- chrom\_amplim, [15](#), [24](#), [26](#), [48](#)
- chrom\_asym, [17](#), [27](#), [42–44](#), [68](#)
- chrom\_bline, [6](#), [8–10](#), [19](#), [24](#), [26](#), [30](#), [48](#)
- chrom\_deriv, [20](#), [22](#), [26](#)
- chrom\_derlims, [21](#), [24–26](#), [64](#)
- chrom\_detect, [3–5](#), [17](#), [19](#), [23](#), [26–28](#), [32–38](#),  
[44](#), [47–50](#), [58](#), [69](#), [70](#)
- chrom\_export, [26](#)
- chrom\_icf, [27](#), [28](#), [29](#), [49](#), [50](#), [59](#), [60](#), [67](#)
- chrom\_plot, [31](#), [57](#)
- chrom\_res, [27](#), [33](#)
- chrom\_retf, [27](#), [34](#), [35](#), [36](#), [37](#)
- chrom\_sepf, [27](#), [34](#), [36](#)
- chrom\_skim, [27](#), [37](#), [49](#), [50](#), [60–62](#), [67](#), [68](#)
- chrom\_smooth, [24](#), [26](#), [41](#), [48](#)
- chrom\_tpa, [17](#), [19](#), [42](#)
- chrom\_tplate, [13](#), [15](#), [27](#), [44](#), [67](#), [68](#)
- chrom\_visc, [14](#), [15](#), [27](#), [45](#), [47](#)
- chrom\_visc (browse\_visc), [11](#)
- chrom\_width, [24](#), [26](#), [42](#), [47](#)
- cprint, [49](#)
- cubspline (linspline), [63](#)
  
- dtprep, [4](#), [5](#), [27](#), [31](#), [41](#), [49](#), [58](#)
  
- egh\_penalty, [31](#)
- emg\_penalty, [31](#)
- etg\_penalty, [31](#)
- exgc1, [50](#)
- exgc2, [51](#)
- exlc1, [51](#)
- exlc10, [52](#)
- exlc11, [52](#)
- exlc2, [53](#)
- exlc3, [53](#)
- exlc4, [54](#)
- exlc5, [54](#)
- exlc6, [55](#)
- exlc7, [55](#)
- exlc8, [56](#)
- exlc9, [56](#)
- exskim, [41](#)
- ext\_lcqc, [57](#), [66](#)
  
- fastchrom\_bline, [17](#), [28](#), [30](#), [31](#), [34–38](#), [41](#),  
[45](#), [49](#), [50](#), [57](#)
- find\_peaks, [58](#)
  
- gs\_penalty, [31](#)
- gskim, [41](#)
  
- icf\_EGH, [31](#)

icf\_EMG, [31](#)  
icf\_ETG, [31](#)  
icf\_GS, [31](#)  
icf\_plot, [28](#), [30](#), [31](#), [59](#)  
init\_egh, [31](#)  
insect, [61](#)  
integ, [31](#), [41](#)  
integ\_plot, [38](#), [40](#), [41](#), [61](#)  
intercept, [61](#)  
intercept (slope), [73](#)  
is.even, [62](#)  
is.odd (is.even), [62](#)  
  
lcqc\_render, [27](#)  
lcqc\_render (render\_defs), [67](#)  
linspline, [63](#)  
  
markmerge, [70](#)  
mean, [42](#)  
  
noise\_plot, [22](#), [64](#)  
  
optim, [28](#), [30](#)  
  
pdrop, [41](#)  
peak\_hw, [31](#), [44](#)  
peakfind, [25](#), [26](#), [32](#), [70](#)  
peakmark, [26](#)  
  
read\_shim, [27](#), [65](#)  
read\_shimadzu, [65](#)  
render\_defs, [67](#), [67](#), [68](#)  
  
sadmin\_infchk, [69](#)  
simlc1, [70](#)  
simlc2, [70](#)  
simlc3, [71](#)  
simlc4, [71](#)  
simlc5, [72](#)  
simlc6, [72](#)  
simlc7, [73](#)  
skimcomp, [41](#)  
slope, [61](#), [73](#)  
smooth\_whit, [6](#), [74](#)  
supp\_pars, [74](#)  
  
tskim, [41](#)  
  
z\_optim, [25](#), [26](#), [75](#), [77](#)  
z\_thres, [15](#), [16](#), [22](#), [75](#), [76](#), [76](#)