First, let's start with the equation that you are fitting to. This is defined in the lmfit docs:

GaussianModel

class **GaussianModel**(*independent_vars=['x'], prefix='', nan_policy='raise', **kwargs*)

A model based on a Gaussian or normal distribution lineshape.

The model has three Parameters: *amplitude*, *center*, and *sigma*. In addition, parameters *fwhm* and *height* are included as constraints to report full width at half maximum and maximum peak height, respectively.

$$f(x; A, \mu, \sigma) = \frac{A}{\sigma\sqrt{2\pi}} e^{[-(x-\mu)^2/2\sigma^2]}$$

where the parameter *amplitude* corresponds to A, *center* to μ, and *sigma* to σ. The full width at half maximum is $2\sigma\sqrt{2\ln 2}$, approximately 2.3548σ.

For more information, see: https://en.wikipedia.org/wiki/Normal_distribution

Parameters:

- **independent_vars** (list of str, optional) – Arguments to the model function that are independent variables default is ['x']).
- **prefix** (*str, optional*) – String to prepend to parameter names, needed to add two Models that have parameter names in common.
- **nan_policy** (*{'raise', 'propagate', 'omit'}, optional*) – How to handle NaN and missing values in data. See Notes below.
- ****kwargs** (*optional*) – Keyword arguments to pass to `Model`.

There is a sneaky line in here that is easy to miss, and that I think is the source of your confusion:

> Parameters *fwhm* and *height* are included as constraints to report full width at half maximum and maximum peak height, respectively.

So, when you fit your data to the model you can extract *both* the `height` and `amplitude` parameters despite only the `amplitude` being optimised. This is because the two are related - to find out where height comes from, we can dive into the lmfit source code for the GaussianModel, and then from there piece together how height and amplitude are related.

```python
class GaussianModel(Model):
    r"""A model based on a Gaussian or normal distribution lineshape.

    The model has three Parameters: `amplitude`, `center`, and `sigma`.
    In addition, parameters `fwhm` and `height` are included as
    constraints to report full width at half maximum and maximum peak
    height, respectively.

    .. math::

        f(x; A, \mu, \sigma) = \frac{A}{\sigma\sqrt{2\pi}} e^{[{-{(x-\mu)^2}/{{2\sigma}^2}}]}

    where the parameter `amplitude` corresponds to :math:`A`, `center` to
    :math:`\mu`, and `sigma` to :math:`\sigma`. The full width at half
    maximum is :math:`2\sigma\sqrt{2\ln{2}}`, approximately
    :math:`2.3548\sigma`.

    For more information, see: https://en.wikipedia.org/wiki/Normal_distribution

    """

    fwhm_factor = 2*np.sqrt(2*np.log(2))
    height_factor = 1./np.sqrt(2*np.pi)

    def __init__(self, independent_vars=['x'], prefix='', nan_policy='raise',
                 **kwargs):
        kwargs.update({'prefix': prefix, 'nan_policy': nan_policy,
                       'independent_vars': independent_vars})
        super().__init__(gaussian, **kwargs)
        self._set_paramhints_prefix()

    def _set_paramhints_prefix(self):
        self.set_param_hint('sigma', min=0)
        self.set_param_hint('fwhm', expr=fwhm_expr(self))
        self.set_param_hint('height', expr=height_expr(self))

    def guess(self, data, x, negative=False, **kwargs):
        """Estimate initial model parameter values from data."""
        pars = guess_from_peak(self, data, x, negative)
        return update_param_vals(pars, self.prefix, **kwargs)
```

```python
def height_expr(model):
    """Return constraint expression for maximum peak height."""
    fmt = "{factor:.7f}*{prefix:s}amplitude/max({}, {prefix:s}sigma)"
    return fmt.format(tiny, factor=model.height_factor, prefix=model.prefix)
```

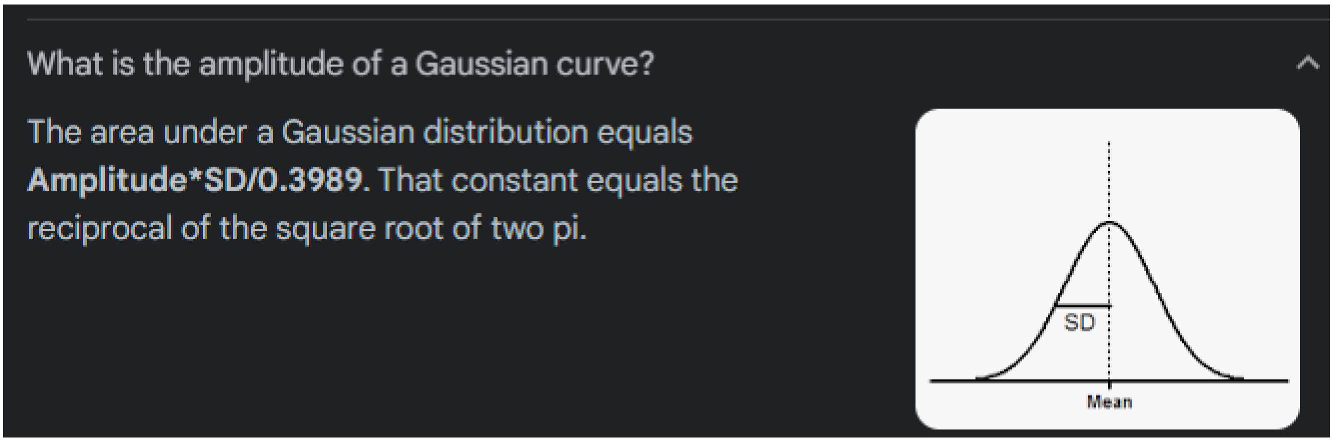
Combining these two things together leaves us with

```
height_factor = 1. / np.sqrt(2*np.pi) #recipricol of the square root of two pi
height = height_factor * amplitude / sigma
```

and rearranging to solve for amplitude gives

```
amplitude = height * sigma / height_factor
```

Now, if that looks vaguely familiar, that's because it is effectively the same as the area formula that you were trying to use!

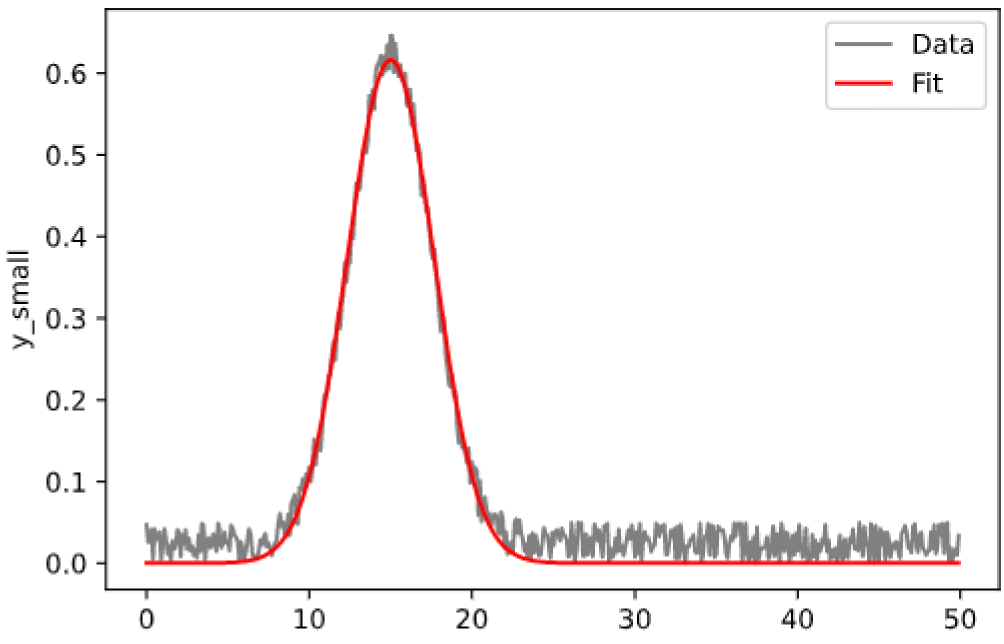


where, in this case SD=`sigma`.

So - long story short, the *amplitude* parameter that you are fitting using lmfit's gaussian model is in fact the area (composed of the sigma and height, and assuming the baseline is at 0). The reason you were getting strange results with your code is you were using this already-transformed parameter and transforming it *again* using the area formula. In your example, I'm guessing the `amplitudes` are quite similar between the two peaks, but the `sigma` of the green peak will be HUGE (it's very broad) while the `sigma` of the blue peak comparatively smaller. What you end up with is multiplying the area of the green peak by LOTS and the blue peak by not so much - resulting in the apparently large green peak area compared to the blue.

Now, just in case you don't believe me (totally your perogative!) then I ran through an example where I fit some simulated data with python and derived the area by fitting and integrating, and also compared it with an online calculator by reverse-engineering the fitted gaussian model. You can play with the code here or see a summary here.

Here is my simulated data fitted to an lmfit gaussian model:



Remember this is the lmfit eqn for the gaussian model:

$$f(x; A, \mu, \sigma) = \frac{A}{\sigma\sqrt{2\pi}} e^{[-(x-\mu)^2/2\sigma^2]}$$

And substituting my fitted parameters into the lmfit equation in the online calculator to determine the area between 0 and 50 via integration:

$$\text{area } f(x) = \frac{4.12}{2.67\sqrt{2\pi}} e^{\frac{-(x-15.0)^2}{14.2578}}, [0, 50]$$

Go

Notice the A parameter (4.12) is effectively the same as the calculated area (4.119):

Show Steps

Area under the curve $f(x) = \frac{4.12}{2.67\sqrt{2\pi}} e^{\frac{-(x-15)^2}{14.2578}}$ on interval $[0, 50]$: $4.11998\ldots$

Steps

Area under a curve definition — *Hide Definition*

The area under a curve is the area between a curve $f(x)$ and the x – axis on an interval $[a, b]$ given by $A = \int_a^b |f(x)|dx$

Apply the area formula: $\int_0^{50} \frac{4.12}{2^{\frac{1}{2}} \cdot 2.67\pi^{\frac{1}{2}}} \left| e^{-\frac{1}{14.2578}(x-15)^2} \right| dx$ — *Show Steps*

Solve $\int_0^{50} \frac{4.12}{2^{\frac{1}{2}} 2.67\pi^{\frac{1}{2}}} \left| e^{-\frac{1}{14.2578}(x-15)^2} \right| dx$: $4.11998\ldots$ — *Show Steps*

The area is:

$= 4.11998\ldots$

And just to confirm it is actually calculating the curve we think it is:

« Hide Plot

Plotting: $\frac{4.12}{2.67\sqrt{2\pi}}e^{\frac{-(x-15)^2}{14.2578}}$ between 0 and 50