A plain-language guide

LiPridic

How the model is built,
and how we know it is real

Written for readers with no modelling background.
Every term is defined the first time it appears.

Start here

What LiPridic is for

Lipid nanoparticles carry RNA into cells. The component that decides whether they work is the **ionizable lipid** — a molecule with a charged head and oily tails. Change the head or the tails and the particle behaves differently. There are more possible lipids than anyone can make.

The usual way to find a good one is to make a lot of them and test each. That is slow and expensive. LiPridic takes a different path: measure a set of lipids carefully, then build a **model** — a mathematical rule — that connects what a lipid *looks like* to how it *behaves*. Once the rule exists, you can ask it about lipids you have not made.

This document explains how that rule gets built, how we check it is not fooling us, and what it can and cannot tell you. No mathematics is assumed.

The two questions LiPridic answers

Question	What the model predicts	How well this works
Will it assemble properly?	Apparent pKa, particle size, polydispersity, encapsulation efficiency	Well. These are physical properties and the structure largely determines them.
Will it deliver RNA?	Potency in cells and in animals	Less well. Biology intervenes. This is an open question, not a solved one.

The one term you need: apparent pKa

A lipid's head group can pick up a positive charge when the surroundings turn acidic. The **pKa** is the acidity at which half the molecules have picked up that charge. “Apparent” means measured on the assembled particle rather than on the molecule alone — the two differ by roughly two to three units. It matters because the particle needs to be neutral in the bloodstream and charged inside the cell, and a pKa near 6.5 achieves both. Most of this document uses pKa as the worked example.

Step one

What the model eats

A model cannot read a chemical structure. It reads numbers. So the first job is turning every lipid into a row of numbers called **descriptors**.

A descriptor is any number you can calculate from the structure alone, without making the compound. How greasy is it. How heavy. How many nitrogen atoms can hold a charge. How long are the tails. These are free — a computer derives them from the drawing in milliseconds.

Descriptor	In plain words	Why it might matter
cLogP	How greasy, on a calculated scale	Greasier lipids pack differently
Molecular weight	How heavy	Tracks overall size
TPSA	How much of the surface is water-friendly	Balance of head against tails
Ionizable N count	How many nitrogens can take a charge	More charge sites, more RNA held
Spacer length	Carbons between head and backbone	Shifts pKa by about a unit
Tail carbon total	Length of the oily part	Affects assembly and pKa
Unsaturation	Number of double bonds in the tails	Changes how fluid the tails are
Linkage type	Amide or ester bond	Decides how fast the body breaks it down

Measured values are descriptors too

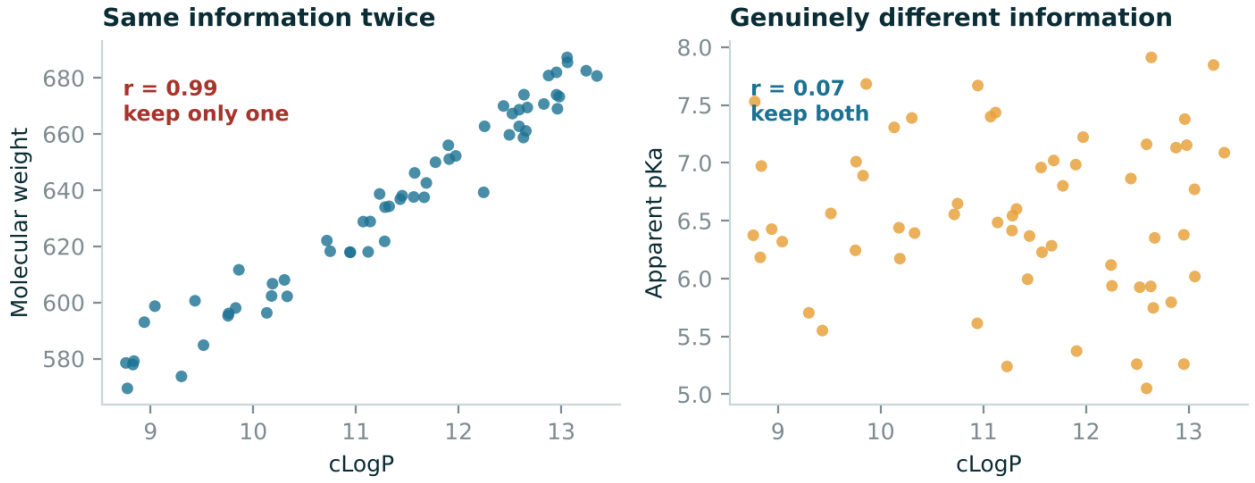
Once a lipid has been made and formulated, its measured properties can also feed the model. Measured apparent pKa, particle size and encapsulation efficiency all become inputs when the target is potency. The distinction matters: calculated descriptors are available before synthesis, measured ones only after.

> **Formulation parameters count as inputs as well**
>
> The same lipid behaves differently at different N/P ratios or lipid mole ratios. LiPridic records those alongside the structure. If they vary across the dataset they enter the model; if they are held constant they carry no information and are dropped. Either way they must be recorded, or a formulation effect gets mistaken for chemistry.

Step two

Throwing away descriptors on purpose

More inputs sounds better. It is not. Two descriptors that always move together carry one piece of information between them, and feeding both in makes the model unstable without making it smarter.



Left: molecular weight rises almost perfectly with cLogP, because in this library only tail length varies. They are one descriptor wearing two hats — keep one. Right: cLogP and apparent pKa move independently, so both earn a place.

This step happens before any measurement exists

It only compares descriptors against each other, never against the answer. That means it can run the day the library is designed, and it will tell you whether your planned set of compounds is secretly measuring the same thing eight different ways — while there is still time to change the design.

Choosing which survivor to keep is a chemistry decision, not a statistical one. LiPridic uses a fixed priority list: cLogP over molecular weight, measured pKa over calculated pKa, charge density over raw nitrogen count. Fixed, so the answer does not drift between runs.

Why not just let the computer pick the best descriptors?

Because “best” means “most related to the answer”, and if you let the model choose its inputs by looking at the answer, then grade itself on the same data, it grades its own homework. The score comes out high and means nothing. In LiPridic, anything that consults the answer happens strictly inside the testing procedure described later.

Step three

Two models, doing different jobs

LiPridic fits more than one model on purpose. They are not competitors; each answers a question the other cannot.

Model A — least squares, the one you can read

Ordinary least squares is the line of best fit, extended to several inputs at once. It assumes the answer is a weighted sum:

```
potency = a×(pKa) + b×(cLogP) + c×(charge density) + … + constant
```

It finds the weights that make the total error as small as possible. That is the whole method.

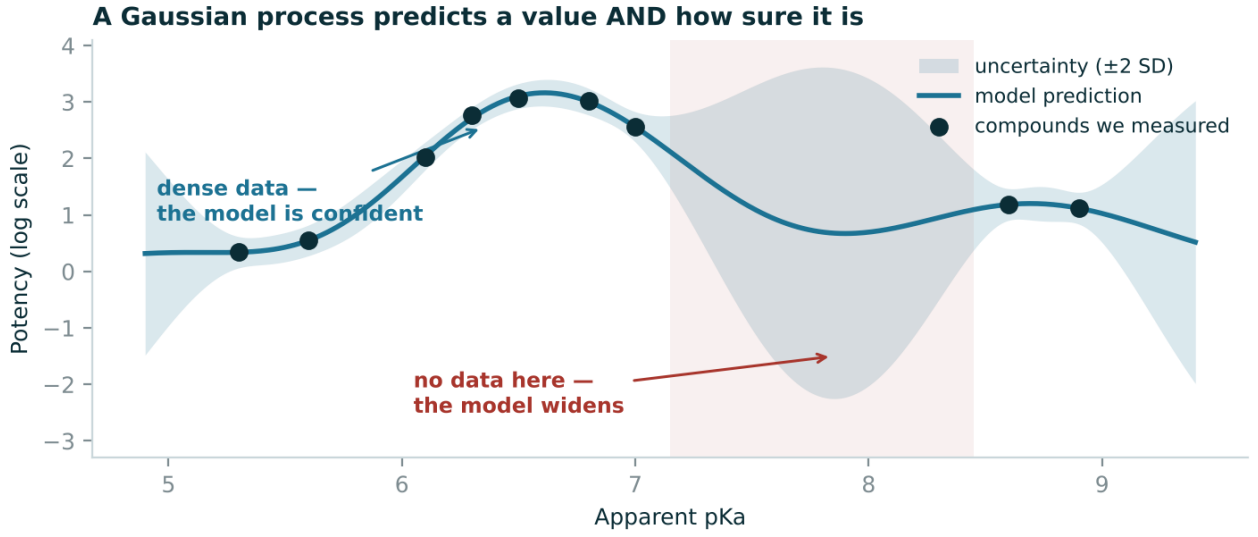
Its value is that every weight is a sentence. *Each unit of apparent pKa above 6.4 costs 0.8 units of potency.* A chemist can check that against what they already know. You also get, free, an honest statement of how uncertain each weight is — which tells you whether an effect is real or noise.

Its limitation is that it can only draw straight lines. If potency peaks at a middle pKa and falls off either side, a straight line through that shape finds nothing. You can add a curve term, but each addition costs you budget — a point we return to.

Model B — Gaussian process, the one that knows what it doesn't know

A Gaussian process makes one assumption: **lipids with similar descriptors behave similarly**. To predict an unmade compound, it takes a weighted average of every compound you did measure, weighting near neighbours heavily and distant ones barely at all.

The important part is what comes out alongside the prediction.



The line is the prediction; the shaded band is the model's own uncertainty. Where measured compounds sit close together the band is tight. Where there is a gap the band balloons — the model is explicitly telling you it is guessing there.

Why that band is the most useful output in the whole system

It turns the model into an experimental planner. A wide band on an unmade compound means measuring it would teach the model a lot. A tight band with a high prediction means it is probably worth making for its own sake. Every new measurement redraws the whole picture, so the list of what to make next updates itself.

Model C — gradient boosting, used as a ruler

Gradient boosting builds hundreds of small yes/no decision trees in sequence, each one correcting the errors of the ones before. *Is the pKa below 6.2? If yes, is cLogP above 14? Then predict low.* Add them all up and you get a prediction.

Because it splits rather than multiplies, it finds peaks, thresholds and combinations automatically. But hundreds of trees are not something you can write in a paper — there is no equation at the end.

So LiPridic does not use it to predict. It uses it to measure. Run it beside least squares and compare:

What you see	What it means	What to do
Both score about the same	The relationship really is roughly linear	Report the least-squares model. Done.
Boosting scores much higher	There is a peak or an interaction the straight line is missing	Go back and add the right curve or combination term
Both score near zero	The descriptors do not capture this property at all	That is a finding. Report it honestly.

Step four

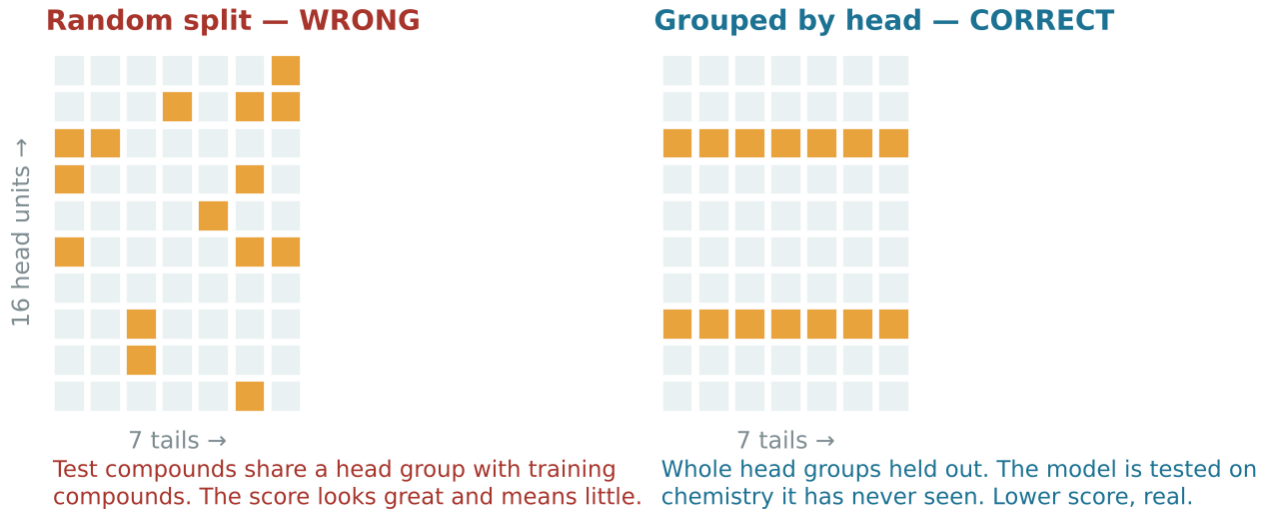
Proving the model is not fooling us

Any model can memorise the data it was shown. The question is whether it has learned anything that transfers. LiPridic runs three independent checks, and all three must pass.

Check 1 — hide whole head groups, not random compounds

The standard approach is to hide a random fifth of the data, train on the rest, and see how well the hidden fifth is predicted. In a combinatorial library that is badly misleading.

Here each head group appears in several compounds that differ only in the tails. Hide them at random and the model sees near-identical siblings of every hidden compound during training. It is being asked to recognise something it has already met.



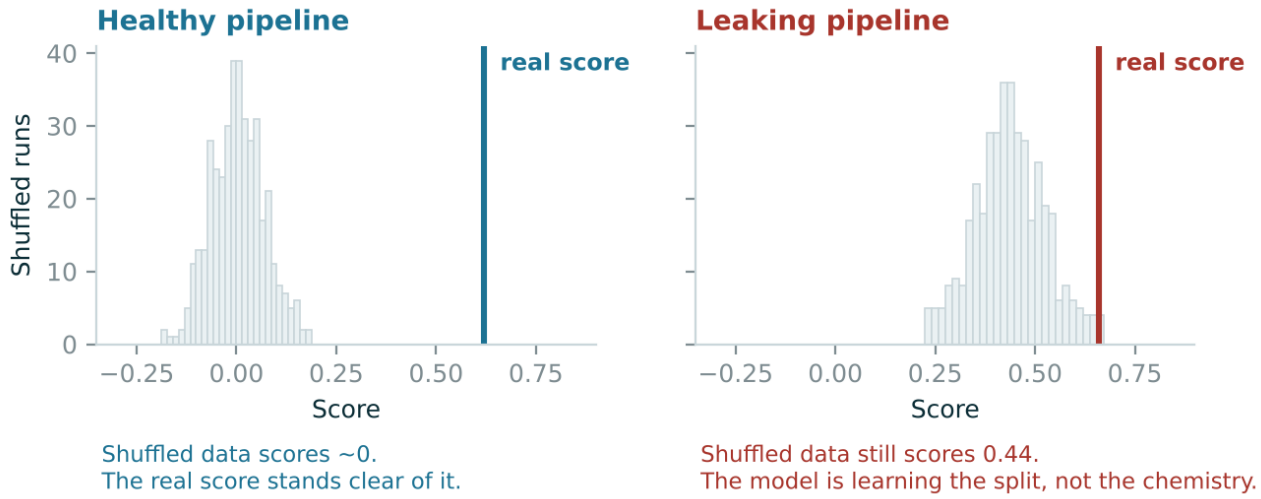
Each square is one compound: head groups down the side, tails across. Orange squares are held back for testing. On the left, random selection leaves siblings on both sides of the split. On the right, whole head groups are held out, so the model faces genuinely new chemistry.

Expect the correct number to look worse

A grouped score of 0.6 is a stronger result than a random-split score of 0.95, and the two are not comparable. Published lipid models often report the random-split kind. Whenever a score appears, LiPridic states which procedure produced it — otherwise the comparison gets made on the wrong terms.

Check 2 — shuffle the answers and make sure the model fails

Take the measured values and scramble them so they no longer belong to any compound. Run the entire procedure again. A healthy pipeline should now score about zero, because there is genuinely nothing to learn. Repeat fifty times and look at the spread.



Left: scrambled data scores around zero and the real score stands well clear. The model has found something. Right: scrambled data still scores 0.44, so the real score of 0.66 is mostly machinery, not chemistry. Something in the pipeline is leaking.

This check has to wrap the *whole* procedure — descriptor selection included. Shuffling only at the final step misses exactly the leak it exists to catch.

Check 3 — a locked holdout, set aside before anything starts

A small number of compounds are flagged at the very beginning and never enter any model, ever. They are measured like everything else and then ignored until the end.

The reason is subtle. Once a model starts suggesting what to make next, the data stops being a fair sample — it concentrates wherever the model was unsure. Every internal score then flatters itself. The locked holdout is the one measurement untouched by that, and it only works if it is chosen before you start looking.

> **The holdout is chosen to cover the edges**
>
> Not at random. The compounds are picked to span the full range of expected pKa, including the extremes, because the edges are exactly where models fail and exactly where a random draw would leave you blind.

Step five

Reading the results

Output	What it is	How to read it
Coefficient	The weight on one descriptor in the least-squares model	“Each extra unit of X changes the answer by this much.” The sentence you can publish.
Confidence interval	The range the coefficient plausibly lies in	If it straddles zero, the effect is not established. Report the interval, not the point.
R^2	Fraction of the variation the model explains, 0 to 1	0.6 means it captures most of the pattern and misses some. Never quote it alone.
RMSE and MAE	Typical size of the model's error, in real units	RMSE 0.25 on pKa means predictions are usually within about a quarter of a unit.
Error bar on a prediction	The Gaussian process's own uncertainty	Wide means the compound sits far from anything measured. Treat it as a guess.
Y-randomisation score	How well the model does on scrambled data	Should be near zero. If it is not, nothing else on the page can be trusted.

Why R^2 alone is not enough

R^2 says how much of the spread you explained, which depends on how spread out the data was to begin with. RMSE says how wrong you typically are, in units a chemist can act on. A model can have an unimpressive R^2 and still be useful if its errors are small, and an impressive R^2 while being useless where it matters. LiPridic always reports both, plus the correlation coefficient.

And why performance is reported in bands

An overall score averages across the whole range. If a model is excellent between pKa 6 and 7 and hopeless outside, the average can still look respectable — and the average is not what you use when designing a new compound. LiPridic reports the score separately for each band of pKa, so a blind spot cannot hide inside a good headline number.

Honesty section

Three limits that do not go away

These are properties of the problem, not flaws in the software. Stating them up front is cheaper than having them found later.

Limit 1 — the number of compounds is not the number of data points

A library built by crossing head groups with tails has fewer independent observations than it has compounds. Every compound sharing a head group tells you about that same head group. For any descriptor that only varies between head groups, the real sample size is the number of head groups — not the number of compounds.

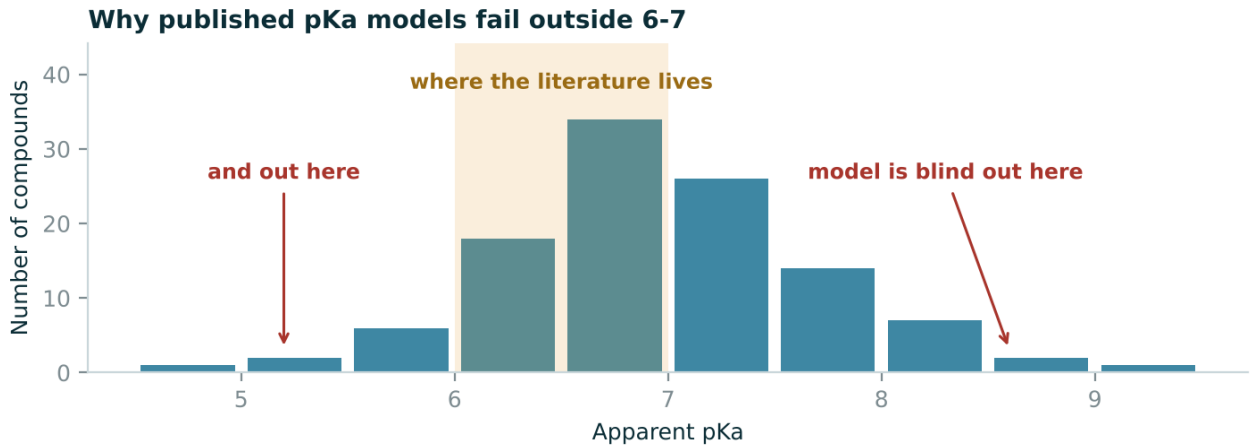
This sets a hard ceiling on how many effects the model can estimate. Making more compounds from the same building blocks does not raise it. Only more distinct building blocks do. LiPridic reports this effective number for every descriptor and warns when one is resting on too few.

Limit 2 — nearly identical lipids can behave completely differently

Published work on large lipid datasets found that most pairs of structurally near-identical lipids differ substantially in delivery — in some cases by a hundredfold or more, from a single atom. The field calls these cliffs.

No model built on structural similarity can predict a cliff, because the structures are similar by definition. This is why potency prediction tops out where it does: the best published deep-learning model, trained on roughly ten times more compounds, explains about half the variation in one cell line and a quarter in another. Expect numbers in that neighbourhood, not better.

Limit 3 — models are confident exactly where they should not be



Published lipid data clusters between pKa 6 and 7, because that is the useful window and therefore what gets published. A model trained on that pile predicts confidently inside it and badly outside — and gives no warning.

This is not hypothetical. The leading published pKa model reports close agreement inside 6–7 and severe deviation outside, and in its own prospective round two of three designed lipids came out above pKa 7 when the model had placed all three in range.

LiPridic handles this two ways. Every prediction is flagged as inside or outside the range the model was trained on. And the recommended design deliberately includes compounds expected to land outside the optimal window — the ones nobody publishes — because they are what anchor the edges of the curve.

In one page

The whole process

	Stage	What happens
1	Describe	Every designed lipid becomes a row of calculated numbers. No synthesis needed.
2	Reduce	Descriptors that duplicate each other are collapsed. Target never consulted.
3	Lock the holdout	A spread of compounds is set aside permanently, before any fitting.
4	Measure	Synthesis, formulation, assays. Replicates kept separately. Reference lipid in every run.
5	Fit	Least squares for readable coefficients; Gaussian process for uncertainty; boosting as a ruler.
6	Check	Grouped validation, shuffled-answer test, per-band scores. All three must pass.
7	Predict	Unmade compounds ranked by predicted value and by uncertainty.
8	Test	Make the compounds the model picked. This is the only real proof.

The last row is the one that matters

Everything before step 8 is a model fitting data it has already seen. Only a prospective round — predicting compounds that do not yet exist, making them, and checking — distinguishes a model from a curve drawn through points. It is also far cheaper than doubling the library.

Reference

Glossary

Term	Meaning
Descriptor	A number calculated from a structure, used as a model input
Target	The thing being predicted (pKa, size, potency)
Collinear	Two descriptors that move together and carry the same information
Coefficient	The weight the model puts on one descriptor
Cross-validation	Hiding part of the data, training on the rest, testing on the hidden part
Grouped validation	The same, but hiding whole families of related compounds
Y-randomisation	Scrambling the answers to confirm the model then learns nothing
Holdout	Compounds set aside permanently and never used for fitting
Overfitting	Memorising the training data instead of learning a transferable pattern
Applicability domain	The region of chemistry the model was trained on, inside which it can be trusted
Prospective validation	Predicting unmade compounds, making them, and checking the prediction
Activity cliff	A pair of near-identical structures with very different behaviour

LiPridic · a personal project. This document describes methodology only and contains no experimental data.