Supplemental Materials

Molecular Biology of the Cell

Pryor et al.

Supplement to Pryor et al, 2015

Domain Reconstruction Algorithm:

Overview

One of the major undertakings of this algorithm is to extract static spatial data from dynamic trajectory data. To accomplish this task, we devised a ranking system that sorts all recorded points of an individual trajectory into two groups: confined (slow moving) or free (unobstructed). The confined points are what would be considered to be in a confinement zone, while the free points are considered to be "free" on the membrane.

Trajectory Analysis

To rank the points in the SPT trajectories, we first calculate the corresponding jump size (displacement) for time steps corresponding to a selected number of frame intervals (1, 2, 5, 10, 20, 50, 100, 200). The time step is applied moving forward through the trajectory as well as backwards through the trajectory. Covering this range of step sizes accounts for "holes" that may be in the trajectory data due to experimental conditions such as blinking of the quantum dots or quantum dots temporarily moving out of the focal plane. These jump sizes are calculated for each trajectory in a set of comparable SPT files, then aggregated by step size.

Once the jump size distributions have been compiled, the individual points from each trajectory are ranked. For every point, the relative rank is calculated by comparing the point's jump size to all the other points' jump sizes for a specific step size. The jump sizes for a specific step size are sorted in order from smallest to largest. The rank of a specific point is where it falls in that order, and is converted into a relative (percentage) rank. This is repeated across all points for each step size.

Some points may not have a score for all 16 step sizes due to the aforementioned holes in the trajectories. To account for this, a weighted average is used to determine the overall rank (score) for each point. First, for each point, the forward and backward ranks for a step size are combined. The combined ranks for each step for that point are then averaged together; this is the final rank for that point.

Cluster Analysis

We rely on distance based hierarchic clustering. This approach had been widely used in the literature, ranging from ecology and genomics to receptors on the cell membrane (Espinoza et al, 2012). Our method is a modified version of that developed by Espinoza and coworkers for TEM images of receptors labeled with gold nano-particles.

Given N points $\{P_1,P_2,\ldots,P_N\}$ in a plane, with coordinates $\{(x_1,y_1),\ldots(x_N,y_N)\}$, we want to partition them into mutually exclusive groups or clusters in a way that reflects their proximity or similarity to each other. This is not a clearly defined notion and the appropriate clustering method should be ultimately determined by the experimental context. Here, we use the "slow" points identified from jump size distributions as indicators of an underlying physical structure (such as lipid rafts); therefore the notion of proximity defined by physical distance to the closest members of the cluster is more appropriate considering, for instance, an average distance to the entire cluster (this is the idea underlying K-means clustering).

We construct clusters by comparing the distance $d(P_1,P_2)=\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$ between points to a reference length L, sometimes called a "scale". Two points A, B are in the same cluster if their distance $d(A,B)\le L$; we denote this relationship by $A \sim_L B$. We extend the relationship by transitivity: if $A \sim_L B$ and $A \sim_L C$, then $B \sim_L C$. It is easy to see that the procedure will induce a partition of the set of points. (The $\sim_L$

relation is symmetric and transitive, therefore it is an equivalence relation in the mathematical sense; the clusters are the corresponding equivalence classes.)

The partition into clusters is unique for a given set of points and length scale L. Any two points in a cluster can be linked by a connected path consisting of line segments of length $\leq L$ connecting points from the same cluster (Figure S1A).

Contour Drawing

In order to build a geometric area (shape or footprint) around a given cluster, we start with the union of the circles of radius *L/2* centered on all members of the cluster. We assume the cluster resulted from hierarchic clustering with distance parameter *L,* so any member of the cluster must be reached from any other member through a sequence of segments connecting cluster points, such that no individual segment is long than *L*. The connection graph, shown in Figure S1A, is constructed by putting an edge between all pairs of points with distance $\leq L$.

To straighten the boundary of the region defined this way, we extend the area by adding rectangles of height L along the contour of the connection graph (double lines Figure S1C). The reconstructed region is the reunion of the inside of the contour graph, and the circles and rectangles around the vertices and edges of the contour (shaded, respectively grey areas on Figure S1D).

The contour graph or "tight contour" for a cluster of points is defined by the sequence of boundary points and the segments that connect them. The list is constructed by adding new points to the contour, based on the existing points and a reference direction. As the contour is built, it circles around the points in a counter-clockwise direction, so that all the interior points are to the left hand side. When the process is finished, the last point in the contour is identical to the first one.

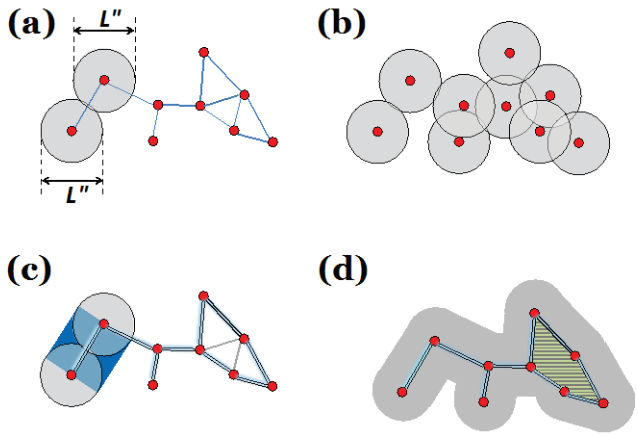

Figure S1: Clustering Algorithm Walkthrough. (a) The points in a cluster form a connected graph, where edges connect points whose distance is less than the length scale L. Circles of diameter L centered on two points intersect if and only if the points are connected in the sense described above. (b) We want to define the footprint based on the reunion of all the circles of diameter L, centered on the points in the cluster. (c) We first identify the outer contour (sometimes a skeleton, with no interior) of the cluster graph (double blue shaded lines). We 'pad' the area defined by the circles by adding rectangles along the edges of the contour graph. (d) The reconstructed region is the reunion of the inside of the contour graph (if any), and the circles and padding rectangles around the vertices and edges of the contour graph.

<u>Contour Building Algorithm</u>

1. Start with the rightmost point of the group; set the reference direction pointing to the right. (Any point on the convex hull is acceptable; the reference direction needs to be pointing toward the outside of the hull.)
2. Add new points:
    a. Identify all the points in a circle of radius *L* centered on the last point added (the current point); these are the candidate points.
    b. Draw line segments from the current point to each candidate point. If this intersects a segment in the already identified part of the contour, discard respective candidate point.

    c. Order the remaining candidates by the clockwise angle from the reference direction to the segment connecting the current point to the candidate; choose the candidate with the smallest angle and add it to the list.
    d. Set reference direction to point from the newly added point back to the previously added point
3. The process terminates when the same *segment* is added to the contour. The same point may be visited twice, in opposite directions. Upon successful termination, the last point in the list is the same as the first one.

NOTE: The contour defined this way is not unique, but the algorithm always returns a contour that is a refinement of the convex hull of the points, is not self-intersecting, and contains no edges longer than L.

Contour Inflation

The tight contour or contour graph defines the reconstructed region. This contour is often lacking an "inside", and can have sections that are just a single chain of points. The rationale of padding is to represent the area "of influence" of each point in the cluster – roughly, the set of geometric locations that are closer to this cluster than to any points that are not part of the cluster.

The padding adds two types of elements to the core contour and its interior:

1. A rectangle of height $L/2$ on the outer side of each edge of the tight contour graph
2. A sector of a circle of radius $L/2$ at each vertex with a positive (convex) angle

The contour inflation algorithm constructs a second contour (the outer or padded contour) that encloses the first one. Similarly to the tight contour, the padded contour defined as a polygonal line; the vertices of the polyline are all auxiliary points, and are not elements of the cluster.

The contour inflation algorithm proceeds along the inner contour, and builds the outer contour parallel to it, adding one or more points to account for each vertex of the inner contour. If the angle at the respective vertex is positive (convex), we add the corners of the padding rectangles and points on the arc of the padding circle centered on the vertex; if the angle is negative (concave), then we only add one point, namely the intersection of the two padding rectangles. The perimeter and area of the final shape are given by the length and enclosed area (if any) of the tight contour, plus easily calculable additions due to the padding procedure. Below we describe the algorithm we employed. This algorithm also relies on a reference direction, which corresponds to the segment preceding the current vertex (with the direction defined by that of the algorithm – counterclockwise in our case).

<u>Contour Inflation Algorithm</u>

1. Start at a point on the tight contour. Moving counter-clockwise along the contour, set the reference direction to that of the incoming edge in the counter-clockwise direction, and the current direction to the outgoing edge.
2. The elements of the new contour that are added to represent the current vertex are set based on the angle between the incoming and outgoing directions.
    a. If the angle is positive (left turn, convex vertex), then add a sector of a circle of radius $L/2$, centered on the current vertex, and delimited by the radii perpendicular to the incoming and the outgoing edges. The sector is discretized as a sequence of points starting with the end the radius orthogonal to the incoming edge and ending at the other one. At least one intermediate point is added at the end of the radius that is the symmetry axis of the sector.
    b. If the angle is negative (right turn, concave vertex), we only add the point where the two adjacent rectangles intersect. This point is also on the symmetry axis (bisector) of the angle on the inner contour.
3. The algorithm proceeds along the vertices of the inner contour, adding points to the outer contour for each of them. The process terminates when the starting vertex is reached, and is about to be passed in the same direction as in the first iteration.

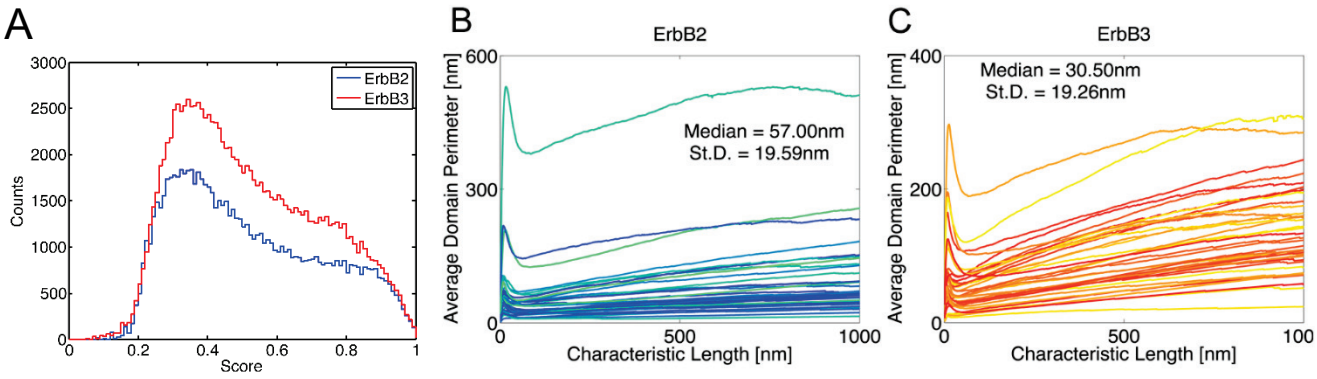

Figure S2. Compiled ranks for all the points from SPT data for ErbB2 and ErbB3. A) The biomodal distribution for both species is independent of QD label pairs. Here, the QD labels are switched by comparison to Figure 2B in main text (erbB2 QD-655; erbB3 QD-585). B) and C). Plots of characteristic length studies. The average domain perimeter was calculated over a range of characteristic lengths for ErbB2 (B) and ErbB3 (C). The minimum of the average domain perimeter was used to determine the optimal characteristic length that defines the maximum distance.

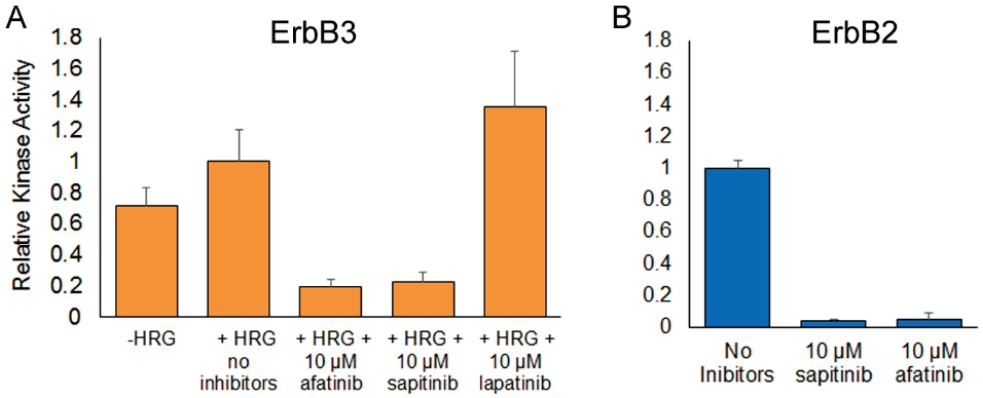

Figure S3. Effects of ErbB inhibitors on ErbB kinase activity. (A) Results of an *in vitro* kinase assay using immunoprecipitated ErbB3 isolated from CHO stably transfected with ^{HA}ErbB2 + ErbB3-mCitrine; cells were incubated with or without 2 min stimulation with 12 nM HRG. Isolated protein was incubated with 10 μM inhibitors (afatinib, sapitinib, or lapatinib) for 15 minutes at 30^{o}C prior to the addition of ATP. Phosphorylation of the EAY substrate was detected using a pan-phospho-tyrosine antibody, PY20-HRP and EAB substrate. (B) Results of an *in vitro* kinase assay on immunoprecipitated ErbB2 isolated from resting CHO ^{HA}ErbB2 cells. Inhibitors were added as in A.

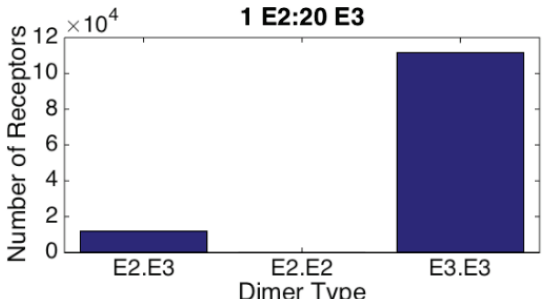

Figure S4. Dimer type distribution after 120 seconds, where ErbB2 is limiting. A BioNetGen version of our model was run for 120 seconds with a ratio of 1:20 ErbB2:ErbB3 receptors. Due to the low level of ErbB2 receptors, ErbB3 receptors are largely sequestered in homodimers.

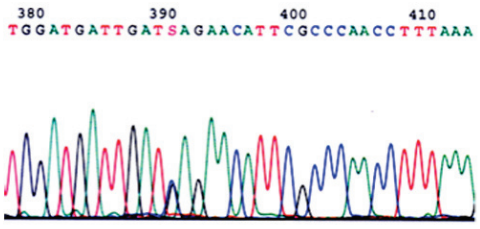

Figure S5. An amino acid substitution of E933Q results from a single point mutation in one of two ErbB3 alleles in the widely-used SKBR3 breast cancer cell line.

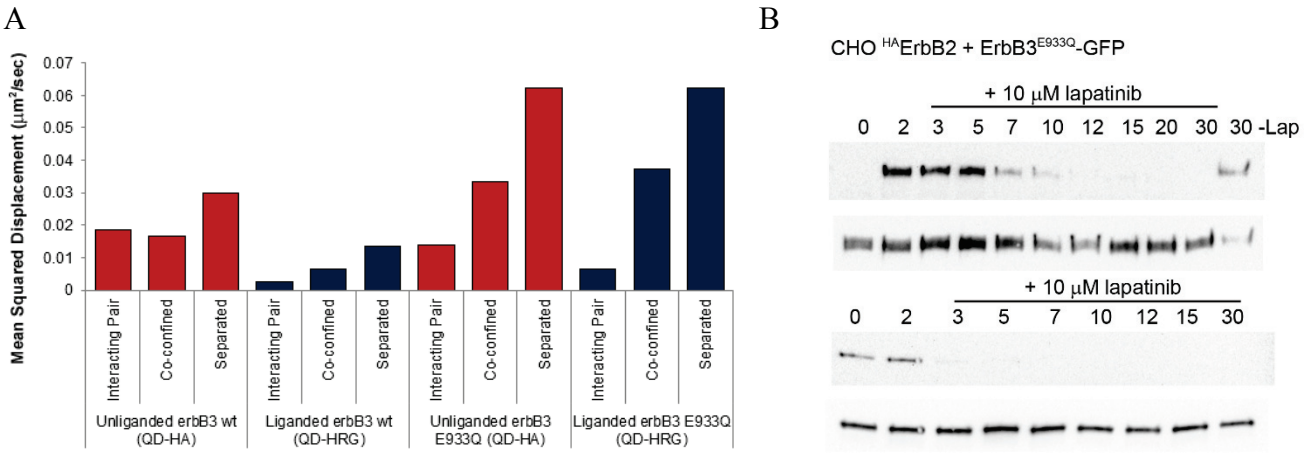

Figure S6. Characterization of the ErbB3 E933Q mutation. A.) State-dependent mean squared displacement of ErbB3wt or ErbB3E933Q receptors was calculated from single particle tracking data. Although ErbB3E933Q receptors have a faster mean squared displacement over all, they show similar slowed diffusion when pairs are interacting especially with ligand. B.) Normalized ErbB3 PY1289 phosphorylation levels after 2 minute stimulation followed by lapatinib treatment (10 μM) in CHO ErbB3^{wt} versus CHO ErbB3^{E933Q} were plotted over time. As in Figure 3, phosphorylation levels of both receptors were set to 1 for the two minute time point after HRG stimulation. Points were fit to a one-phase exponential decay curve to determine the dephosphorylation half-life.

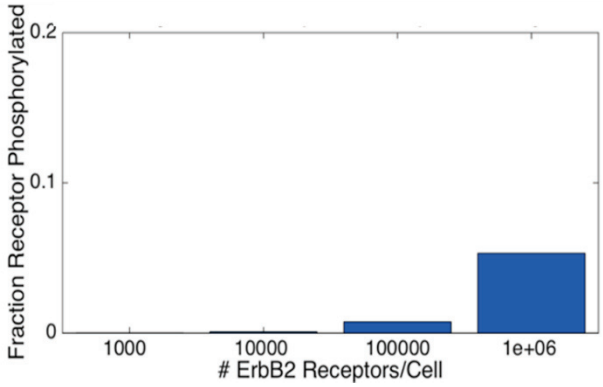

Figure S7. Simulation results with higher levels of ErbB2 alone show increased ErbB2 phosphorylation indicating that overexpression of ErbB2 leads to the formation of active ErbB2 homodimers. ErbB2 is able to overcome the low on-rate for homodimer formation at high density.