**Appendix A. Supplementary data**

**Table 1**. Dataset description

|  |  |  |  |
| --- | --- | --- | --- |
| **Data name** | **Data format** | **Coordinate system** | **Data source** |
| China provincial boundary | Polygon shp | GCS\_WGS\_1984 | Socioeconomic Data and Applications Center (SEDAC) |
| Nanjing boundary | Polygon shp | GCS\_WGS\_1984 | NSTI-Geodata.cn |
| DEM | Geotiff | GCS\_WGS\_1984 | NSTI-Geodata.cn |
| Nanjing 1:250, 000 inhabitants  | Point shp | GCS\_WGS\_1984 | NSTI-Geodata.cn |
| Distribution of employed population at district level | Point shp | GCS\_WGS\_1984 | National Bureau of Statistics of China (NBS) |
| Nanjing’s Sentinel-2A MSI remotely sensed data | Raster | UTM\_50N | Remote Pixel |
| Nanjing’s land use and land cover (LULC) data | Raster | Krasovsky\_1940\_Albers | NSTI-Geodata.cn |
| Road network in Nanjing | Polyline shp | WGS\_1984\_UTM\_Zone\_50N | Openstreetmap (OSM) |
| Distribution of prefecture-level nature reserves | Polygon shp | WGS\_1984\_World\_Mercator | NSTI-Geodata.cn |
| Annual mean precipitation distribution data of Nanjing | Polygon shp | GCS\_WGS\_1984 | NSTI-Geodata.cn |
| Location distribution of per capita GDP | Polygon shp | GCS\_WGS\_1984 | NSTI-Geodata.cn |

Note: Nanjing’s LULC and remotely sensed data used 2015 as the base year.

**Appendix B. LEAM model calibration and validation**

* 1. **Model calibration**

Calibration is the key to identifying areas for re-development and development. LEAM uses a stochastic greedy algorithm (SGA) to measure connectivity to city and population centers for current land use cells and uses the outcome to identify highly probable re-development land areas among existing developed land. Generation of new development probabilities in LEAM includes population centers, employment centers, and other attractions (can be defined by users) as attractants or repellents to new developments. The original data types for those attractions are point data, although in LEAM their influence is not constrained as points. For example, people may prefer to live close to commercial centers for shorter commuting time, material needs, and job opportunities. Thus, it is necessary to include spatial dependencies of accessibility to important attractions in the considerations.

The conventional spatial matrix uses direct neighborhood or straight-line distance as the cutoff. However, in our case, it is more appropriate to use the shortest travel time from each cell to each attractor as a proxy for spatial weight. The typical method to calculate shortest travel times uses the Dijkstra gravity algorithm (Goodman et al. 2016). In this case, our densely connected raster of n cells (more than 1 million cells in the Stockholm region) means a computational complexity of O(n2) for each attractor (Goodman et al. 2016). This suggests that the calculation time of the Dijkstra algorithm increases by a power of two for total number of cells, which means that the calculation would be computationally expensive and take a long time for each of the (10 or so) attractors used in our model.

An alternative to Dijkstra is the greedy algorithm approach. A greedy algorithm has a much smaller computational complexity of O(n) for the same shortest travel time calculation. A greedy algorithm sends agents out from a given attractor in random patterns until it discerns the most efficient pathway. The approach finds local optima instead of Dijkstra’s more overall optima, which means that the algorithm sacrifices performance for computational efficiency. To maintain the balance between performance and efficiency, LEAM uses a parallel SGA to determine attractor travel times. In this way hundreds of greedy algorithms are run simultaneously. Each is assigned a randomized decision rule and a chance to “jump” out of its local optimization routine to find a more globalized optimization (Viswanathan et al., 2011). Figure 1 presents summary logic of the SGA process for finding the shortest (TT) path from one population center k to other cells.



**Figure 1**. Summary logic of the stochastic greedy algorithm (SGA) approach for shortest travel time calculations.

As noted, the computational complexity for the SGA algorithm is O(TN), but in this case N is the number of iterations a modeler specifies greedy algorithms to take and T is the distance cutoff that is applied (since one cell is 30x30 m, the modeler can specify that locations more than 1,000 steps (30,000 m) away from a population center do not matter). Note that N and T are significantly smaller than the total number of cells, n, and thus SGA has a much smaller computational cost than Dijkstra. Further, N processes of randomized agent dispatching have no dependencies, so it is possible to parallel the N processes, reducing the computational time to O(TN/C) (where C is the number of threads available for parallelization). Several attractors can be run simultaneously, which can potentially run almost as fast as one greedy algorithm.

To consider the size of population centers for accessibility, we use transform the travel time into an attractiveness score by the following equation:

 $a\_{ik}=\frac{p\_{k}}{c\_{ik}+l}$ (1)

where $a\_{ik}$ represents the attractiveness score from center *k* for cell *I*; $p\_{k}$ is the total population of center *k*; $c\_{ik}$ is the travel time cost from cell I to center *k*; and *l* is a normalization factor. In this proposed equation, *l* is set to a fixed time (20 minutes) that the travel time from a center of less than 20 minutes does not make a major difference in terms of attractiveness from the center for the cell. Through the calculation of the relationship between attractiveness and the land use change probability, we found that the areas with higher accessibility value to population centers in Nanjing have higher probability of land use change. Using this relation calibration, we build a probability map for new urban built-up land for Nanjing. Also, the final built-up land allocation is made by the highest probability as well as the local city development demand.

* 1. **Model validation**

For validation, variable weighting, and calibration, we use a modified multi-resolution fitting process. Additionally, Deal et al. (2017) argue that PSS-based scenario planning processes and outcomes can be improved by including the ability to do multi-directional temporal analyses, such as “re-casting” from a point in time in the past to the current condition (Deal et al., 2017). By re-casting, modelers can compare past predictions with current “ground-truth” conditions. This comparison requires a reliable validation method. It enables modelers to pinpoint parts of the model that need improvement, to weigh variables correctly, and to support and validate future forecasts. For planners, it can also validate (or invalidate) assumptions made regarding past policies.

We use a modified multi-resolution fitting process to communicate model goodness-of-fit. Model evaluations are performed using an expanding window to gradually degrade the resolution of the comparison, yielding information not contained in single resolution methods. The method can provide a clearer picture of how a model performs in each smaller, specific spatial zone, without losing information as in conventional cell-by-cell level statistical approaches. We use zonal constructs that are based on statistical patterns, using general goodness-of-fit criteria and applied to different (multiple) scaled zones simultaneously to calibrate and validate modeled outcomes. The approach allows goodness-of-fit to be examined at different spatial resolutions and mitigates the arbitrariness of both visual comparisons and spatial boundary selection in a way that is both objective and reasonable.

**Reference**

Deal, B., Pan, H., Timm, S., & Pallathucheril, V. (2017). The role of multidirectional temporal analysis in scenario planning exercises and Planning Support Systems. *Computers, Environment and Urban Systems*, *64*, 91–102.

Goodman, L., Lauschke, A., & Weisstein, E. W. (2016). Dijkstra’s Algorithm. *MathWorld - A Wolfram Web Resource*. Retrieved at http://mathworld.wolfram.com/DijkstrasAlgorithm.html

Viswanathan, V., Sen, A. K., & Chakraborty, S. (2011). Stochastic Greedy Algorithms. *International Journal on Advances in Software*, *4*(1&2), 1-11.