PECAS - for Spatial Economic Modelling

Theoretical Formulation

System Documentation Technical Memorandum

Template Materials Providing Basic Descriptive Components

JD Hunt and JE Abraham
HBA Specto Incorporated

Calgary, Alberta

Table of Contents

"Today's scientists have substituted mathematics for experiments, and they wander off through equation after equation, and eventually build a structure which has no relation to reality."

Nikola Tesla, 1857-1943

1. Introduction

1.1. Overview of PECAS

PECAS is a generalized approach for simulating spatial economic systems. It is designed to provide a simulation of the land use component of land use transport interactive modelling systems.

PECAS stands for Production, Exchange and Consumption Allocation System. Overall, it uses an aggregate, equilibrium structure with separate flows of exchanges (including goods, services, labour and space) going from production to consumption based on variable technical coefficients and market clearing with exchange prices. It provides an integrated representation of spatially distinct markets for the full range of exchanges, with the transport system and the development of space represented in more detail with specific treatments.

Flows of exchanges from production to exchange zones and from exchange zones to consumption are allocated using nested logit models according to exchange prices and transport generalized costs (expressed as transport utilities with negative signs). These flows are converted to transport demands that are loaded to networks in order to determine congested travel utilities. Exchange prices determined for space inform the calculation of changes in space thereby simulating developer actions. Developer actions are represented at either (a) the level of individual land parcels or grid cells using a microsimulation treatment or (b) the level of land use zones using an aggregate flow treatment. The system is run for each year being simulated, with the travel utilities and changes in space for one year influencing the flows of exchanges in the next year.

1.2. Purpose of This Document

This document is designed to describe the theoretical formulation of PECAS. It outlines the concepts and the mathematical structure of the system and the processes used to provide representation of the relevant elements in the real-world system being simulated.

This document does not describe the PECAS software and it does not provide specific guidance on how to either develop or run a PECAS model. Such descriptions and guidance are provided in other system documentation technical memoranda. In particular, users are encouraged to check the HBA Specto website https://www.hbaspecto.com and also refer to the PECAS Software Implementation and User Guide, System Documentation Technical Memorandum 2: for descriptions of the PECAS software and how to implement it on a computer, and instructions on how to run a PECAS model, specify inputs and obtain outputs.

1.3. Content of This Document

This document describes the concepts used in PECAS. It indicates the approaches used to represent time, space and the behaviour of the system elements, including the mathematical formulae and their linkages into a system.

The four basic system modules forming a complete spatial economic model system with PECAS are identified and described briefly. The treatments of time, locations, land and developed space are outlined.

The two basic system modules included in PECAS are the Activity Allocation Module (AA Module) and the Space Development Module (SD Module). There are two forms of the SD Module: a disaggregate version (labeled SD Module) and an aggregate version (labeled SD-A Module). The theoretical approaches, typical configurations, mathematical formulations, solution algorithms and derivations for these modules are presented. The correspondence between the variable names and their labels in the software are also indicated.

2. Basic Model System Modules

PECAS includes two basic modules that are linked together with two other basic modules to provide a representation of the complete spatial economic system.

The set of four basic modules includes:

- Space Development Module (SD Module): This is one of the two PECAS modules. It represents the actions of developers in the provision of different types of developed space where activities can locate, including the new development, demolition and re-development that occurs from one point in time to the next. This developed space is typically floorspace of various types and is called 'space' in the PECAS framework.

- Activity Allocation Module (AA Module): This is the other of the two PECAS modules. It represents how activities locate within the space provided by developers and how these activities interact with each other at a given point in time.

- Transport Model (TR Module): This is one of the 'non-PECAS' modules. It represents the transport system connecting locations, including at a minimum a transport network, the transport demands that load onto this network (as a result of the economic interactions represented in the AA Module) and the congested times and costs for interactions between locations arising with the loading of these demands. A standard, aggregate transportation planning model is likely to be sufficient. PECAS ships with a simple travel model called "PECAS Assign" which can be used to calculate network congestion, but this is rarely used, as most agencies have a more sophisticated travel model.

- Economic Demographic Aggregate Forecasting Model (ED Module): This is the other of the 'non-PECAS' modules: It is some form of model or approach used to develop aggregate economic forecasts for the study area being modeled.

Typically, these forecasts include projected numbers of households or population by category and employment by type (as indications of expected economic activity) for specific points of time in the future. This model or approach may be able to adjust its forecasts in response to information from the AA and SD modules – as is represented in the descriptions included here – or it may provide a static set of forecasts. It may even be the case that there is no model per se that is available, merely the forecast values for the study area. It is also possible to use an extended form of the PECAS AA Module to develop such aggregate forecasts – by making some specific assumptions about the relative contributions to the study-area economy from inside and outside the study area. For the descriptions included here, all of these possibilities are included in the single 'ED Module' designation that is used.

3. Treatment of Time

The four basic modules listed above are linked together with information flows as shown in Figure 1.

This linked system works through time in a series of discrete, fixed steps from one point in time to the next, with the AA Module running at each point in time and the SD Module considering the period from each point in time to the next. In general, the fixed steps can be of any duration, but 1 year time steps are recommended since they allow an appropriately quick response of land developers in the SD Module to the space prices established in the AA Module.

Ideally, the transport model (TR Module) used to calculate the congested travel times and associated transport utilities is run for each year, after the AA Module has been run for that year. If the overall model run times are too long and travel conditions are relatively stable, the TR Module can be run less often to save computation time.

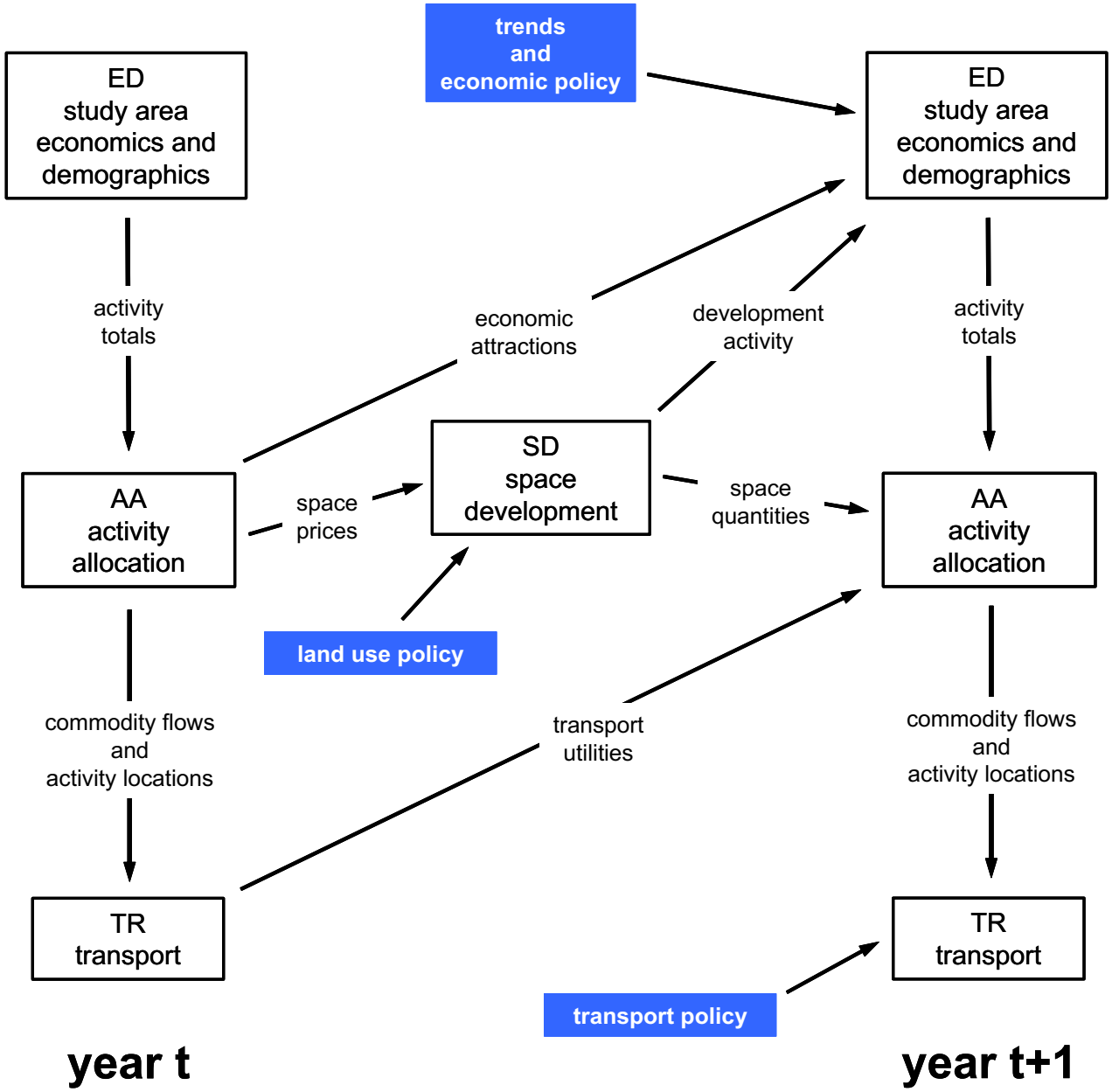

Figure 1: Modules and Information Flows Simulating Temporal Dynamics

4. Treatment of Locations, Land and Developed Space

The study area is organized into a set of land use zones (LUZ). In the AA Module activities locate in these zones and puts flow between them. Ideally these zones match the transport zones (TAZ) used in the TR Module or are aggregations of whole numbers of adjacent TAZ. The connectivity among the LUZ is based on the representation provided by the TR Module, where the TR Module establishes congested network times and costs and associated transport utilities between TAZs that the AA Module uses in its consideration of the interactions between the LUZs.

Each LUZ represents a spatial market for goods, services, labour, and space in AA. PECAS models rarely use more than about 1000 LUZs, and LUZs are rarely more finely detailed than what would be considered “neighbourhoods”. In the USA a Census Tract targets 4,000 people, which can be a good size for a residential LUZ. (Census Tracts are often too big in high-employment areas, since they are based on population distribution, not employment distributions.)

The land in each LUZ is further partitioned into smaller cells or parcels. Cells can be formed with a grid pattern over the land, for example there is a world-wide 30m land cover dataset based on Landsat imagery (Liu et al., 2020). More commonly, parcels correspond to actual legal parcels or portions of legal parcels. Undeveloped or agricultural parcels are often too large to represent the complexity of future urban development, so a GIS tool called “parcel cut” is provided with PECAS to assist in dividing these large parcels into smaller individual grid-like development sites. A cadastre dataset often also contains small slivers of parcels, as a real artifact of land subdivision or an artificial artifact of GIS processing – PECAS parcels need to be large enough to hold a development, such as a building or a group of related buildings, so small parcels should be removed or combined.

The term ‘parcel’ is used to refer to both grid cells, undivided parcels, and portions of subdivided parcels in the descriptions below. In the microsimulation version of the SD Module, developed space (called ‘space’) is located on these parcels, with only one type

of space on a given parcel, and the total quantity of each type of space in the TAZ is the sum of the quantities on the parcels in the TAZ, and the total quantity of each type of space in the LUZ is the sum of the quantities of the parcels in the LUZ.

When an activity in the AA Module is located in a zone, it consumes space in the zone, this at rates consistent with the production technology or technologies it is using in the zone. Land is used in the provision of the space in the zone, as an input to the development process as represented in the SD Module.

5. Activity Allocation Module

5.1. Approach

The Activity Allocation Module (AA Module) is an aggregate representation. It concerns quantities of 'activities', flows of 'puts' and markets with aggregate demands and supplies and exchange prices.

Activities are located in LUZ. Activities produce puts and then transport and sell these puts; and they also consume puts after buying them and transporting them. There are different types of activities, including industrial sectors, government and households. Activity quantities can be measured in values (for example, dollars of business repair industrial activity) or numbers (for example, number of households with high income and 2 or less persons). The AA Module allocates the study-area wide quantity of each activity among the LUZ as part of its allocation process.

Puts flow at specific rates from where they are produced as an output to where they are exchanged (from seller to buyer), and then from where they are exchanged to where they are consumed as an input. The word "put" is a generalization of the words "input" and "output" and represent the connections between activities that often flow in physical form in vehicles as trips on the transportation system. Puts are grouped into categories, including different types of goods and services, labour and space. Puts other than space in general flow across zone boundaries. Space is restricted in that it is 'non-transportable' and must be exchanged and consumed in the LUZ where it is produced – which means that the space put categories receive some special additional treatments in PECAS as described further below. Put flows are measured in values per unit time (for example, dollars of management services per year) or numbers per unit time (for example, tonnes of coal per month). The movement of these flows of puts from where they are produced to where they are consumed is the economic basis for travel and transport in the modelling system. It is the travel conditions – the distances, costs, times and associated (dis)utilities by mode – for the movement of these puts that results in the influence of the

transportation system on the interactions among activities and the attractiveness of locations for activities. The AA Module allocates the flows of puts from production location LUZ to exchange location LUZ and from exchange location LUZ to consumption location LUZ, and finds the corresponding set of prices at the exchange location LUZ that clears all markets, as part of its allocation process.

(Puts used to be called “Commodities” and are still often called Commodities in the software implementation of PECAS. The word “Commodity” was problematic for economists, since it has a specific meaning in their field.)

Activities produce puts (as outPUTS) and consume puts (as inPUTS) in the production process according to the technology they use. More specifically, an activity quantity in a given LUZ produces puts at specific rates per unit of activity and consumes puts at specific rates per unit of activity according to the technology being used by the activity. One or more ‘technology option’ alternatives are defined for a given activity. Each of these technology options is a specific vector of production and consumption rates for different puts per unit of the activity, representing a particular technology option for the production process available to the activity. The AA Module allocates the quantity of the activity in each LUZ among these ‘technology options’ as part of its allocation process.

The allocation process in the AA Module uses a three-level nested logit model with a nesting structure as shown in Figure 2.

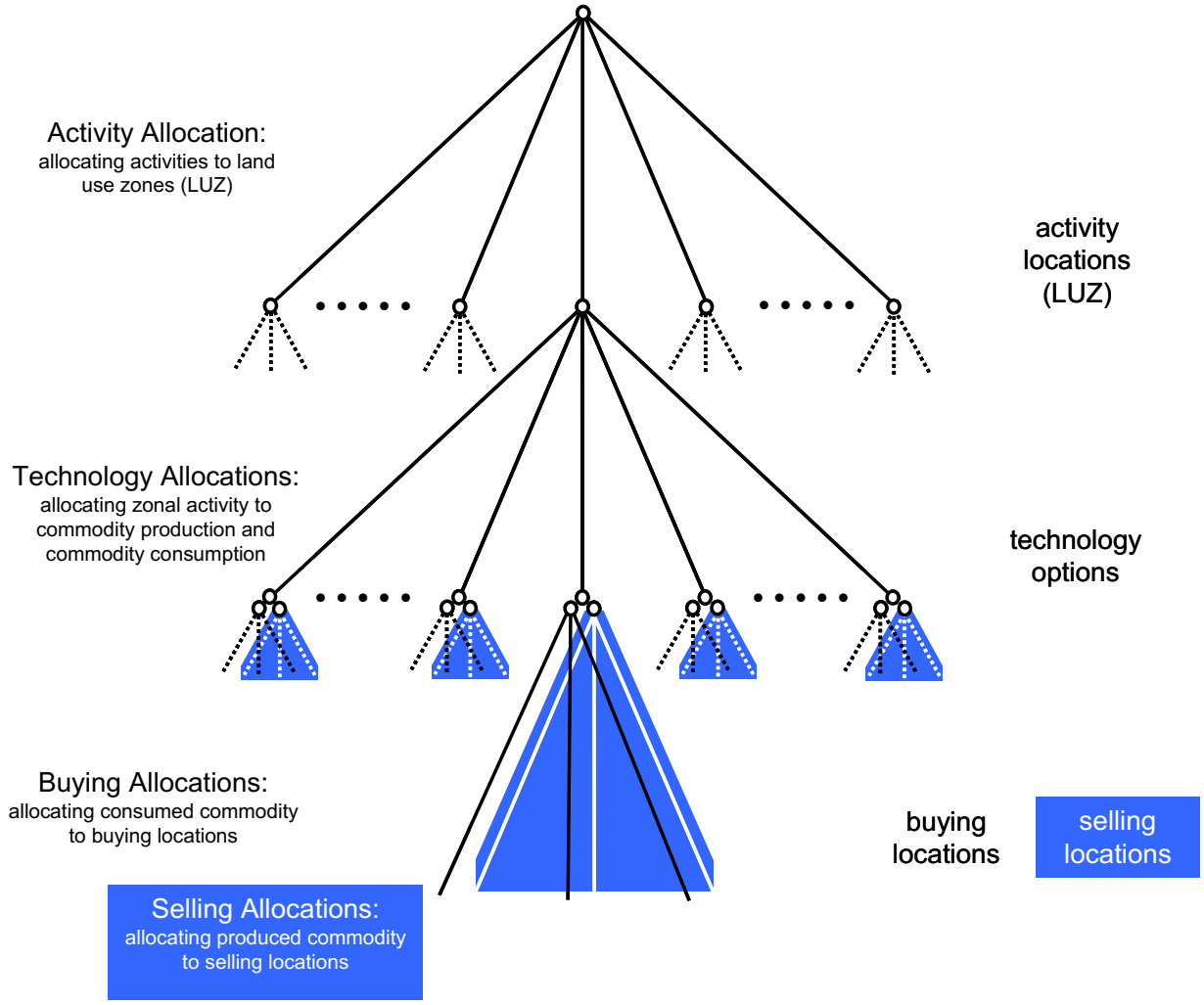

Figure 2: Three-level Nesting Structure Used in Allocations

At the highest level of the nesting structure, the study-area total quantity of each activity is allocated among the LUZ. At the middle level, the quantity of each activity in each LUZ is allocated among the available technology options. At the lowest level, there are two logit allocations for each put in each LUZ: The first is an allocation of the produced quantities among the various exchange locations where they are sold to other activities; the second is an allocation of the consumed quantities among the various exchange locations where they are bought by other activities.

At the lowest level, the utility of each exchange location alternative is influenced by the price at the exchange location, the characteristics for transporting the put to or from the exchange location, and a measure of the size of the exchange location. The composite utility values from these two lowest-level logit models are called the 'buying utility' and the 'selling utility' for the put in the LUZ. They are used as the transportation-related inputs in the middle-level for allocating the activities in the LUZ among the relevant technology options. The composite utility value for the range of technology options considered at the middle-level for an activity in an LUZ is part of the location utilities used at the highest-level.

The spatial aspects of the AA Module allocation process are illustrated in Figure 3. Buying and selling allocations link through the exchange locations to establish put flows from production to nsumption locations in the LUZ.

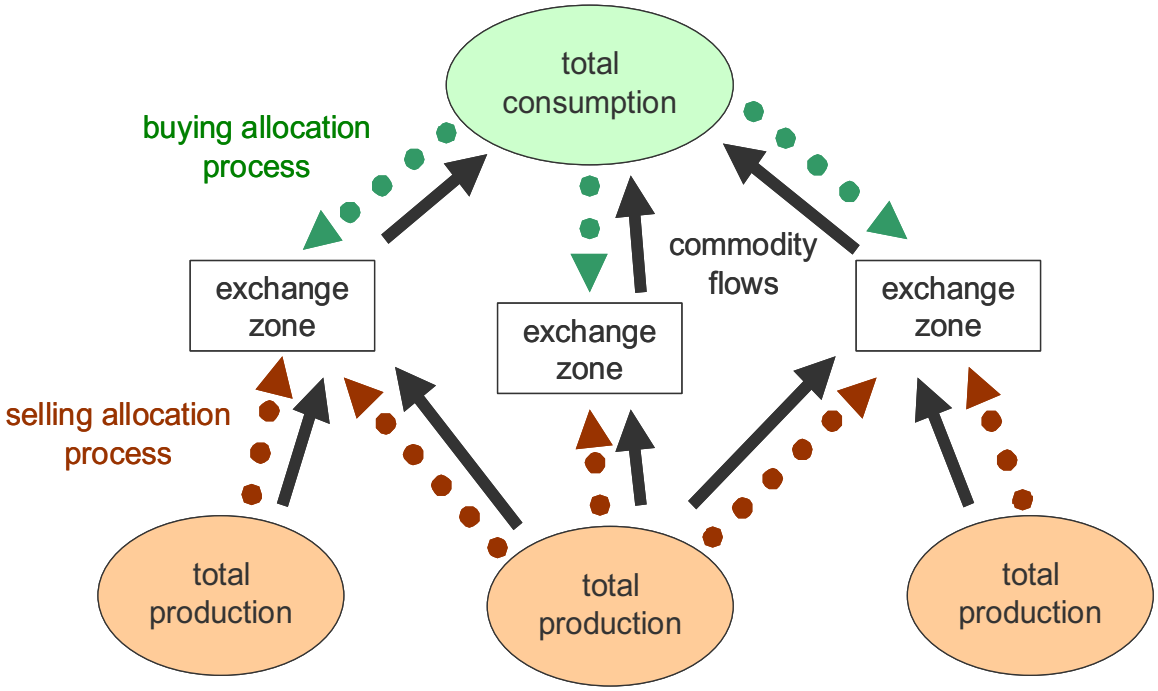

Figure 3: Buying and Selling Allocations Resulting in Put Flows from Production Zone to Consumption Zone Via Exchange Location

The exchange locations are location-specific markets for puts, where sellers sell puts to buyers. Prices are established at exchange locations so that the quantity bought equals the quantity sold – thus the spatial allocation procedure in the AA Module assumes a short-run market equilibrium in puts.

For simplicity and realism, puts can be assigned to be either (a) purchased from the exchange location in the LUZ where they are consumed (e.g. for labour, this occurs in the employment zone, where the labour is exchanged and the price set), or (b) purchased in the exchange location in the LUZ where they are produced (e.g. for retail goods this is at the retail establishment, where the goods are exchanged and the price is set). In this case for each put in each zone, one of the logit allocation models (either the buying or the selling) consists of only one alternative.

There are external market LUZs established for Import Producers and Export Consumers. Importer Producers and Export Consumers are special types of activity that represent the rest-of-the-world market for a particular put. Import Producers produce a variable amount of one particular put, interpolating between two Technology Options. One technology option represents zero supply to the region, effectively selling all their production to other areas of the world rather than the modelled region. Another represents full supply to the region, selling all their production to the model region. Similarly, Exporter Consumers have one technology option set to zero, ignoring the region, and one set to consume all of their needs of a put from within the region. These special activities are almost always constrained to be located only in the external LUZs that represent the rest-of-the-world, so that trade relationships between the region can be represented by flows to external zones in the travel model.

Exogenous amounts of supply and demand can be specified for each exchange as functions. These are often set to non-zero but insignificant values, to aid in the search algorithm by eliminating infinite derivatives.

Table 1: Potential activity categories for activity allocation module

Industry	Government
Agriculture in Office Floorspace	Government in Office Space
Agriculture on Agricultural Land	Government in Government Support Land
Light Industry in Office Floorspace	
Light Industry in Light Industrial Floorspace	
Heavy Industry in Office Floorspace	**Households**
Heavy Industry on Heavy Industrial Land	Low Income Households
Wholesale in Office Floorspace	Medium Income Households
Wholesale in Warehouse Floorspace	High Income Households
Retail in Office Floorspace	
Retail in Retail Floorspace	
Transportation Handling in Office Floorspace	
Transportation Handling in Depot Floorspace	
Other Services in Office Floorspace	
Other Services in Light Industrial Floorspace	
Other Services in Retail Floorspace	
Gradeschool Education in Office Floorspace	
Gradeschool Education on Gradeschool Land	
Post-Secondary Education in Office Floorspace	
Post-Secondary Education on Insititutional Land	

5.2. Typical Categories in Implementation

Table 1 provides a listing of a typical set of activity categories for an implementation of the AA Module.

In general, standard industrial sector categories are used as the starting point for the representation of the business and government components of the economy, with these categories split according to the type of employment and associated space consumed. In general, a given industrial sector is split into blue-collar and white-collar components and associated ‘shop-floor’ and ‘office’ uses in this way so that the allocation process in the model can consider the very different behavior of these two components regarding location, labour use and space type use.

It is not shown in Table 1, but government may be further split according to function, perhaps into services provision, defense provision, health funding, education funding and so on.

Households are typically split according to income, as shown in Table 1, and also size in terms of number of members, not shown in Table 1.

An additional group of activities that can be represented, but are not shown in Table 1, are the investment and saving factors of production generating financial flows that are sources of revenue and expenditure in the economy. Such activities are not essential in an implementation of the AA Module, since they don't require trips on the transportation network, and don't influence the spatial distribution of activities, but can be included to provide a more complete representation of the economic structure of the model area.

Figure 4 provides a summary of the forms of interactions among types of activities and puts for an implementation of the AA Module. Put production is represented in a 'make table format' at the top of the figure; put consumption is represented in a corresponding 'use table format' at the bottom of the figure; puts 'flow' down the columns from where they are produced to where they are consumed. The general nature of the exchange location for each put category and the treatment of imports and exports are also indicated.

	commodities								
	goods	internal management services	services	education services	retail services	domestic services	labour	non-residential space	residential space
producing activities (making)									
industry	goods production by industry	internal management services production by industry	services production by industry	education services production by industry	retail services production by industry				
government	goods produced by government		services production by government	education services production by government					
households						domestic services production by households	labour production by households		
exports	yes		yes		yes		yes		
exchange locations	at production	at consumption	at production	at production	at production	at consumption	at consumption	non-transportable	non-transportable
imports	yes	yes	yes		yes		yes		
consuming activities (using)									
industries	goods consumption by industry	internal management services consumption by industry	services consumption by industry	education services consumption by industry			labour consumption by industry	space consumption by industry	
government	goods consumption by government		services consumption by government	education services consumption by government			labour consumption by government	space consumption by government	
households	goods consumption by households - direct		services consumption by households	education services consumption by households	goods consumption by households - via retail	domestic services consumption by households			space consumption by households

Figure 4 Summary of categories of activities and puts in make and use tables

Labour is included as a put that is produced by households and consumed by economic production activities, with labour in different occupation categories treated as different put categories.

Different types of space (labeled 'floorspace') are consumed by activities and thus are also included as puts in the use table. But there is no production of space by activities in the AA Module. It is the space development (SD) Module running between time periods that adjusts the amount of space supply available in zones, as described below. These amounts of space supply are then input to the AA Module for the next time period, and these amounts are what is available for consumption as fixed quantities in each zone for that next time period when it is considered with the AA Module. Each of these categories of space is 'non-transportable' and thus a given supply of space of a given type must be exchanged and consumed in the same zone where it is produced (or supplied) and thus located.

Such things as water, energy, and air-borne or water-borne emissions can also be included as puts in the make and/or use table structure, which allows the model to determine aggregate demands or impacts arising from activities consistent with the rest of the model simulation.

5.3. Mathematical Formulation

Activity Allocation to Zones

In a given year, the model-wide quantity for each activity is established by the ED Module and input to the AA Module. This 'activity totals' information flow is shown in Figure 1. The AA Module allocates the model-wide quantity for each activity among the LUZ using a logit allocation as follows:

$$W_{a,z} = TW_a \cdot (\exp (\lambda_{l,a} \cdot LU_{a,z}) / \Sigma_{z \in Z} (\exp (\lambda_{l,a} \cdot LU_{a,z}))) \quad (01)$$

with:

$$LU_{a,z} = \alpha_{size,a} \cdot 1/\lambda_{l,a} \cdot \ln [Size_{a,z}] + \alpha_{inert,a} \cdot \ln(PrevW_{a,z} + InertCons_a) + Constant_{a,z} + \Sigma_{v \in V} (\alpha_{a,v} \cdot X_{v,z}) + \alpha_{tech,a} \cdot CUTech_{a,z} \quad (02)$$

where:

z = index representing zones (LUZ);
in software: PecasZonesI(ZoneNumber);

Z = the set of all zones (LUZ);

a = index representing activity categories;
in software: ActivitiesI(Activity);

c = index representing put categories;
in software: CommoditiesI(Commodity);

$LU_{a,z}$ = location utility for a unit of activity a in zone z;
in software: ActivityLocations(LocationUtility);

$W_{a,z}$ = quantity of activity a in zone z;
in software: ActivityLocations(Quantity);

TW_a = model-wide total quantity of activity a;
in software: ActivitiesI(Size);

$\lambda_{l,a}$ = utility function dispersion parameter for allocation of activity a among location zones;
in software: ActivitiesI(LocationDispersionParameter);

$Size_{a,z}$ = representation of relative size of zone z for activity a, indicating the *a priori* expected share of activity a for zone z;
in software: ActivitiesZonalValuesI(SizeTerm) when an input, but usually calculated based on floorspace quantities (see size term discussion in Software User Guide document).

$\alpha_{size,a}$ = utility function coefficient for the sensitivity to size for activity a;
in software: ActivitiesI(SizeTermCoefficient);

$PrevW_{a,z}$ = the proportion of model-wide quantity of activity a in zone z in the previous time period;
in software: ActivitiesZonalValuesI(InitialQuantity);

$InertCons_a$ = Coefficient modifying the sensitivity to the previous portion of activity a in zones, that reduces the importance of the quantity when the previous quantity is small;
in software: fixed at 0 internally, no input included
may eventually include input as ActivitiesI(InertiaTermConstant);

α_{inert} = utility function coefficient for the sensitivity to the previous proportion of activity a in zone z, representing inertia in allocation of activity a;
in software: fixed at 0 internally, no input included
may eventually include input as ActivitiesI(InertiaTermCoefficient);

$Constant_{a,z}$ = utility function alternative specific constant for zone z for allocation of activity a;
in software: ActivitiesZonalValuesI(ZoneConstant) when an input

and: ActivityLocations(LocationSpecificUtility) when an output from constrained run

v = index representing 'other' zonal attributes;
in software: by addition to ActivitiesZonalValuesI(ZoneConstant);

V = the set of all 'other' zonal attributes;

$X_{v,z}$ = one of the 'other' zonal attributes;
in software: by addition to ActivitiesZonalValuesI(ZoneConstant);

$\alpha_{a,v}$ = utility function coefficient for the sensitivity to the 'other' zonal attribute $X_{v,z}$;
in software: by addition to ActivitiesZonalValuesI(ZoneConstant);

$CUTech_{a,z}$ = composite utility associated with the range of technology options for activity a in zone z, defined in Equation 07 and discussed below;
in software: ActivityLocations(TechnologyLogsum)

$\alpha_{tech,a}$ = utility function coefficient for the sensitivity to composite utility associated with range of technology options for activity a;
in software: ActivitiesI(ProductionUtilityScaling) currently is fixed at 1.0;

The types of 'other' zonal attributes that are considered vary depending on the production activity being allocated. For residential activities in particular, these 'other' attributes include representations of various amenities relevant to housing location choice, such as school quality, general noise levels, air quality, open space density, municipal taxation levels and possibly zonal-level income distributions and racial compositions. In the the PECAS software, the influences of these other zonal attributes on location utilities are incorporated by calculating the contribution to location utility in a separate process and then adding this contribution to the location utility constant $Constant_{a,z}$. This

Technology Allocation

The AA Module allocates the quantity of each activity in each LUZ among the available technology options using a logit allocation as follows:

$$Tech_{p,a,z} = W_{a,z} \cdot (\exp (\lambda_{p,a} \cdot UTech_{p,a,z}) / \Sigma_{p \in Pa} (\exp (\lambda_{p,a} \cdot UTech_{p,a,z}))) \quad (03)$$

with:

$$UTech_{p,a,z} = 1/\lambda_{p,a} \cdot \ln [OpSize_{p,a}] + UProd_{p,a,z} + UCons_{p,a,z} \quad (04)$$

and

$$UProd_{p,a,z} = \Sigma_{n \in Np|Rate(p,a,n)>0} \; Rate_{p,a,n} \cdot Scale_{p,a,n} \cdot CUSell_{cn,z} \quad (05)$$

$$UCons_{p,a,,z} = \Sigma_{n \in Np|Rate(p,a,n)<0} \; -Rate_{p,a,n} \cdot Scale_{p,a,n} \cdot CUBuy_{cn,z} \quad (06)$$

where:

p = index representing technology options;
Pa = the set of all technology options available for activity a;
$UTech_{p,a,z}$ = technology utility for a unit of activity a in zone z applying technology option p;
$Tech_{p,a,z}$ = quantity of activity a in zone z applying technology option p;
$\lambda_{p,a}$ = utility function dispersion parameter for allocation of activity a among technology options;
in software: ActivitiesI(ProductionSubstitutionNesting);
$OpSize_{p.a}$ = representation of relative size of application of technology option p for activity a, indicating the a priori expected share of application of technology option p by activity a;
in software: TechnologyOptionsI(OptionSize);
$UProd_{p,a,z}$ = the component of utility arising for a unit of activity a in zone z with production of puts for technology option p;
$UCons_{p,a,z}$ = the component of utility arising for a unit of activity a in zone z with consumption of puts for technology option p;
n = index representing technical coefficients
in software: One column in TechnologyOptionsI
Np = the set of technical coefficients for technology option p
In software: in the columns of TechnologyOptionsI other than the columns named Activity, OptionName or OptionSize
$Rate_{p,a,n}$ = the rate at which put c is produced (if positive) or consumed (if negative) by activity a using technology option p;
*In TechnologyOptionsI entries in columns named **CommodityName** or **CommodityName**:# or Make:**Commodityname** or Make:**CommodityName**:# are taken as is; entries in columns named Use:**CommodityName** or Use:**CommodityName**:# are multiplied by negative 1.*
$Scale_{p,a,n}$ = scaling factor for the utility of $Rate_{p,a,n}$,
in software: Currently fixed at 1.0, to be set in TechnologyOptionsScalingI in a column with the same name as $Rate_{p,a,n}$
cn = Put associated with *n*

in software: In TechnologyOptionsI, Commodity name in column header, possibly prefixed by make: or use: and possibly suffixed by :# (each column must have a unique name)

$CUSell_{c,z}$ = composite utility associated with selling a unit of put c produced in zone z, defined in equation 14 and discussed below
in software: CommodityZUtilities(zUtility) with BuyingOrSelling = S and CommodityName = C

$CUBuy_{c,z}$ = composite utility associated with buying a unit of put c consumed in zone z, defined in equation 18 and discussed below
in software: CommodityZUtilities(zUtility) with BuyingOrSelling = B and CommodityName = C

The composite utility associated with applying the range of technology options for activity a in zone z is determined consistent with equations 03 and 04 as follows:

$$CUTech_{a,z} = (1/\lambda_{p,a}) \cdot \ln (\Sigma_{p \in Pa} (\exp (\lambda_{p,a} \cdot UTech_{p,a,z}))) \quad (07)$$

The composite utility for the range of technology options, $CUTech_{a,z}$, includes the utilities for buying, $CUBuy_{c,z}$, and for selling, $CUSell_{c,z}$, the individual puts that are used and made with those technology options. That is, for each technology option, Equations 05 and 06 sum the utilities $CUBuy_{c,z}$ and $CUSell_{c,z}$ with weights indicating the rates at which they are used, $ConsRate_{p,a,n}$, and made, $ProdRate_{p,a,n}$, as well as any scaling factors $Scale_{p,a,n}$ to establish the utilities for consumption, $UCons_{p,a,,z}$, and production, $UProd_{p,a,z}$, for that technology option. Equation 04 combines these utilities for consumption and production with a size term to establish the utility for that technology option, $UTech_{p,a,z}$. Equation 07 then combines these utilities for the range of technology options to establish the composite utility for the range of technology options, $CUTech_{a,z}$.

The utilities for buying, $CUBuy_{c,z}$, and for selling, $CUSell_{c,z}$, reflect the accessibilities to the puts. Thus, the composite utility for the range of technology options, $CUTech_{a,z}$, combines the accessibilities for buying and selling individual puts consistently with the technology (production and consumption) options for the activity. The location utility (determined in Equation 02) of an activity that consumes a large amount of a put will be

strongly influenced by the composite utility of buying that put, and the location utility of an activity that produces a large amount will be strongly influenced by the composite utility of selling. As such, the last term in Equation 02, $CUTech_{a,z}$, provides overall indications of the accessibility of the zone for the activity consistent with the range of technologies applied by the activity in that zone.

Production and Consumption Quantities

The quantity of put c produced in zone z by activity *a* is the sum of the quantities of put *c* produced by each technology option over the set of technology options applied by activity a, as follows:

$$TPA_{c,a,z} = \Sigma_{p \in Pa} \Sigma_{n \in Pa \mid rate(p,a,n)>0,\ cn=c} Rate_{p,a,n} \cdot Tech_{p,a,z} \quad (08)$$

where:

$TPA_{c,a,z}$ = quantity of put c produced by activity a in zone z using all technology options;
In software:ZonalMakeUse(Amount when Activity=a, MorU=M, ZoneNumber=z)

The quantity of put c produced in zone z by all activities is the sum of the quantities of put c produced over the set of activities, as follows:

$$TP_{c,z} = \Sigma_{a \in A} TPA_{c,a,z} \quad (09)$$

where:

$TP_{c,z}$ = quantity of put c produced by all activities in zone z using all technology options;

Similarly, the quantity of put c consumed in zone z by activity a is the sum of the quantities of put c consumed by each technology option over the set of technology options applied by activity a, as follows:

$$TCA_{c,a,z} = \Sigma_{n \in Pa \mid rate(p,a,n)<0,\ cn=c} -Rate_{p,a,n} \cdot Tech_{p,a,z} \quad (10)$$

where:

$TCA_{c,a,z}$ = quantity of put c consumed by activity a in zone z using all technology options;
In software:ZonalMakeUse(Negative of Amount when Activity=a, MorU=M, ZoneNumber=z)

The quantity of put c consumed in zone z by all activities is the sum of the quantities of put c consumed over the set of activities, as follows:

$$TC_{c,z} = \Sigma_{a \in A} TCA_{c,a,z} \quad (11)$$

where:

$TC_{c,z}$ = quantity of put c consumed by all activities in zone z using all technology options;

Buying and Selling Allocation

The AA Module allocates the total quantity of each put produced in a given LUZ among the exchange locations (where it is sold) using a logit allocation as follows:

$$S_{c,z,k} = TP_{c,z} \cdot (\exp (\lambda s_c \cdot SU_{c,z,k}) / \Sigma_{k \in K} \exp (\lambda s_c \cdot SU_{c,z,k})) \quad (12)$$

with:

$$SU_{c,z,k} = \delta s_{size,c} \cdot 1/\lambda s_c \cdot \ln [XsSize_{c,k}] + \delta s_{price,c} \cdot Price_{c,k} + \delta s_{tran,c} \cdot Tran_{c,z,k} \quad (13)$$

where:

k = index representing exchange locations (LUZ);
K = the set of all exchange locations;
$S_{c,z,k}$ = quantity of put c produced in zone z allocated to be sold in exchange location k (hence is shipped from zone z to exchange location k);
*in software: selling_**CommodityName**.zipMatrix entry with origin=z,destination=k*
$SU_{c,z,k}$ = utility for selling in exchange location k a unit of put c produced in zone z;
$XsSize_{c,k}$ = representation of relative size of exchange location k for selling put c, indicating the *a priori* expected share of put c sold in exchange location k;
in software: ExchangeImportExportI(SellingSize), or calculated as described below
$Price_{c,k}$ = the unit exchange price for put c in exchange location k;
in software: ExchangeResults(Price);
$Tran_{c,z,k}$ = the utility for transporting a unit of put c from zone z to exchange location k, as calculated in Equation 20 below;
$\delta s_{size,c}$ = utility function coefficient for the sensitivity to size when selling put c;
in software: CommoditiesI(SellingSizeCoefficient);
$\delta s_{price,c}$ = utility function coefficient for the sensitivity to price when selling put c;
in software: CommoditiesI(SellingPriceCoefficient);
$\delta s_{tran,c}$ = utility function coefficient for the sensitivity to transport utility when selling put c;
in software: CommoditiesI(SellingTransportCoefficient);
λs_c = utility function dispersion parameter for allocation of selling of put c
in software: CommoditiesI(SellingDispersionParameter).

In the case of selling, the coefficient $\delta s_{price,c}$ is positive.

The selling composite utility for put c (independent of the producing activity) is determined consistent with Equations 12 and 13 as follows:

$$CUSell_{c,z} = (1/\lambda s_c) \cdot \ln \left(\Sigma_{k \in K} \exp (\lambda s_c \cdot SU_{c,z,k}) \right) \quad (14)$$

where:

$CUSell_{c,z}$ = composite utility associated with selling a unit of put c produced in zone z, independent of activity
in software: CommodityZUtilities(zUtility) when BuyingOrSelling=S

The AA Module allocates the total quantity of each put consumed among the exchange zones in a manner that is analogous to the allocation of produced quantities. Specifically, the total quantity of each put consumed in a given LUZ is allocated among the exchange locations (where it is bought) using a logit allocation as follows:

$$B_{c,z,k} = TC_{c,z} \cdot (\exp (\lambda b_c \cdot BU_{c,z,k}) / \Sigma_{k \in K} \exp (\lambda b_c \cdot BU_{c,z,k})) \quad (16)$$

with:

$$BU_{c,z,k} = \delta b_{size,c} \cdot 1/\lambda b_c \cdot \ln [XbSize_{c,k}] + \delta b_{price,c} \cdot Price_{c,k} + \delta b_{tran,c} \cdot Tran_{c,k,z} \quad (17)$$

where:

$B_{c,z,k}$ = quantity of put c consumed in zone z allocated to be bought in exchange location k (hence is shipped from exchange location k to zone z);
*in software: buying_**CommodityName**.zipMatrix entry with origin=k,destination=z*

$BU_{c,z,k}$ = utility for buying in exchange location k a unit of put c consumed in zone z;

$XbSize_{c,k}$ = representation of relative size of exchange location k for buying put c, indicating the *a priori* expected share of put c bought in exchange location k;
in software: ExchangeImportExportI(BuyingSize), or calculated as described below

$Tran_{c,k,z}$ = the utility for transporting a unit of put c from exchange location k to zone z, as calculated in Equation 20 below;

$\delta b_{xsize,c}$ = utility function coefficient for the sensitivity to size when buying put c;
in software: CommoditiesI(BuyingSizeCoefficient);

$\delta b_{price,c}$ = utility function coefficient for the sensitivity to price when buying put c;
in software: CommoditiesI(BuyingPriceCoefficient);

$\delta b_{tran,c}$ = utility function coefficient for the sensitivity to transport utility when buying put c;
in software: CommoditiesI(BuyingTransportCoefficient);

λb_c = utility function dispersion parameter for allocation of buying of put c
in software: CommoditiesI(BuyingDispersionParameter).

In the case of buying, the coefficient $\delta b_{price,c}$ is negative.

The buying composite utility for put c (independent of the consuming activity) is determined consistent with Equations 16 and 17 as follows:

$$CUBuy_{c,z} = (1/\lambda b_c) \cdot \ln (\Sigma_{keK} \exp (\lambda b_c \cdot BU_{c,z,k})) \quad (18)$$

where:

$CUBuy_{c,z}$ = composite utility associated with buying a unit of put c consumed in zone z, independent of activity
in software: CommodityZUtilitiesI(zUtility) when BuyingOrSelling=B

The utility for transporting a unit of put c from any zone j to any zone k, $Tran_{c,j,k}$, is calculated using up to a maximum of three interchange attribute values (that are provided by the TR Module) as follows:

$$Tran_{c,j,k} = \kappa 1_c \cdot IntAtt1_{j,k} + \kappa 2_c \cdot IntAtt2_{j,k} + \kappa 3_c \cdot IntAtt3_{j,k} \quad (20)$$

where:

$IntAtt1_{j,k}$ = value for attribute 1 from land use zone j to land use zone k used to calculate the utility for transporting a unit of put c;
in software: value in column for attribute 1 in row for zone j to zone k in file output from the TR Module as specified using the skim.filename program variable in the aa.properties file;

$IntAtt2_{j,k}$ = value for attribute 2 from land use zone j to land use zone k used to calculate the utility for transporting a unit of put c;
in software: value in column for attribute 2 in row for zone j to zone k in file output from the TR Module as specified using the skim.filename program variable in the aa.properties file;

$IntAtt3_{j,k}$ = value for attribute 3 from land use zone j to land use zone k used to

calculate the utility for transporting a unit of put c;
in software: value in column for attribute 3 in row for zone j to zone k in file output from the TR Module as specified using the skim.filename program variable in the aa.properties file;

$\kappa 1_c$ = utility function coefficient for the sensitivity to attribute 1 when transporting a unit of put c;
in software: CommoditiesI(InterchangeCoefficient1)

$\kappa 2_c$ = utility function coefficient for the sensitivity to attribute 2 when transporting a unit of put c;
in software: CommoditiesI(InterchangeCoefficient2)

$\kappa 3_c$ = utility function coefficient for the sensitivity to attribute 3 when transporting a unit of put c;
in software: CommoditiesI(InterchangeCoefficient3)

An 'exchange regime' is specified for each put c. This exchange regime indicates the spatial nature of the exchanges available for the put. For example, some puts are only exchanged where they are produced – and thus the exchange zone must be the zone of production. The exchange regime for each put c is designated using single-letter code for the variable $ExCh_c$ as follows:

$ExCh_c$ = single-letter code indicating exchange regime for put c;
in software: CommoditiesI(ExchangeType)
with values:

- 'c' = exchanged only in consumption zones (where the seller does all transporting)
- 'p' = exchanged only in production zones (where the buyer does all transporting)
- 'a' = exchanged in any zone (where both buyer and seller may do some of the transporting)
- 'n' = non-transportable (where the put is consumed in the same zone where it is produced)
- 's' = exchanged only in specified zones (both buyer and seller may do some of the transporting, but exchanges occur in only certain zones);
  in software: zones where exchanges occur indicated with 'true' value put in ExchangeImportExportI(SpecifiedExchange)).

Exogenous Supply and Demand

Exogenous supply and demand is established at each exchange using price-sensitive functions. For backwards-compatibility software reasons these are called "imports" and

"exports" at each zone, but imports and exports are no longer represented this way in PECAS. Rather, these are typically set to very small values to aid the software in calculating derivatives and undertaking the Newton-Raphson search procedure described below. The quantities of exogenous supply and demand for a given put in a given exchange zones are determined using:

$$Qi_{c,k} = QiRef_c + \Delta i_c \cdot ([G_i-1]/[G_i+1]) + \mu i_c \cdot (Price_{c,k} - PriceIRef_c) \quad (21)$$

and

$$Qe_{c,k} = QeRef_c + \Delta e_c \cdot ([G_e-1]/[G_e+1]) + \mu e_c \cdot (Price_{c,k} - PriceERef_c) \quad (22)$$

with:

$$G_i = \exp (\eta i_c \cdot (Price_{c,k} - PriceIRef_c)) \quad (23)$$

and

$$G_e = \exp (\eta e_c \cdot (Price_{c,k} - PriceERef_c)) \quad (24)$$

where:

$Qi_{c,k}$ = quantity of put c exogenously supplied to exchange location k;
in software: ExchangeResults(Imports);

$Qe_{c,k}$ = quantity of put c exogenously demanded from exchange location k;
in software: ExchangeResults(Exports);

$Price_{c,k}$ = the unit exchange price for put c in exchange location k;
in software: ExchangeResults(Price);

$QiRef_c$ = quantity of put c exogenously supplied to exchange location when the unit exchange price for put c in exchange zone k is at the reference level $PriceIRef_c$;

in software: ExchangeImportExportI(ImportFunctionMidpoint);

$PriceIRef_c$ = reference price per unit for exogenous supply of put c;
in software: ExchangeImportExportI(ImportFunctionMidpointPrice);

$QeRef_c$ = quantity of put c exogenously demanded from exchange location when the unit exchange price for put c in exchange zone k is at the reference level $PriceERef_c$;
in software: ExchangeImportExportI(ExportFunctionMidpoint);

$PriceERef_c$ = reference price per unit for exogenous demand of put c;
in software: ExchangeImportExportI(ExportFunctionMidpointPrice);

Δi_c = function coefficient for the rate of increase in exogenous supply of put c for exponent term;
in software: ExchangeImportExportI(ImportFunctionDelta);

μi_c = function coefficient for the rate of increase in exogenous supply of put c for linear term;
in software: ExchangeImportExportI(ImportFunctionSlope);

ηi_c = function coefficient for sensitivity to difference in exchange price for put c concerning increase in exogenous supply of put c for exponent term;
in software: ExchangeImportExportI(ImportFunctionEta);

Δe_c = function coefficient for the rate of increase in exogenous demand of put c for exponent term; in software:
in software: ExchangeImportExportI(ExportFunctionDelta);

μe_c = function coefficient for the rate of increase in exogenous demand of put c for linear term;
in software: ExchangeImportExportI(ExportFunctionSlope);

ηe_c = function coefficient for sensitivity to difference in exchange price for put c concerning increase in exogenous demand of put c for exponent term;
in software: ExchangeImportExportI(ExportFunctionEta).

In the case of exogenous supply, the coefficient Δi_c is positive and the coefficient μi_c is positive provided ηi_c is positive.

In the case of exogenous demand, the coefficient Δe_c is negative and the coefficient μe_c is negative provided ηe_c is positive.

If exogenous supply coefficients are specified for puts that are space quantities, they will be ignored (see below for the treatment of space)

Space Quantities

The physical quantities of space are fixed in each zone and non-transportable. These physical quantities are determined for a run of the AA Module for year t by the SD Module in its consideration of the developer actions during the step from year t-1 to year t. The produced supply of a given type of space available as a put in a particular land use zone – the amount that is available to be consumed by activities in the zone – arises from the corresponding physical quantity of the space in the zone. In the short-run situation considered by the AA Module for each year t, landlords make some proportion (and not in general all) of the physical quantity of space available as supply, with the proportion in a zone increasing as the price for the space in the zone increases. The remaining proportion of the physical quantity for each type is unoccupied, or vacant. Expressed as a percentage, this remaining proportion that is unoccupied is typically called the 'vacancy rate'.

The quantity of space type h available as a put in zone z, based on the proportion made available by landlords of the total physical quantity of space of type h in the zone, is calculated as follows:

$$SpcS_{h,z} = SpcQty_{h,z} \cdot SpcPrp_{h,z} \quad (25)$$

and

$$SpcPrp_{h,z} = SpcBas_h + \Delta_h \cdot ([G_h-1]/[G_h+1]) + \mu_h \cdot (Price_{h,z} - PriceBas_h) \quad (26)$$

with:

$$G_h = \exp (\eta_h \cdot (Price_{h,z} - PriceBas_h)) \quad (27)$$

where:

$SpcS_{h,z}$ = quantity of space type h in zone z available as a put and consumed in zone z; there is a put index c that corresponds to the space type index h, so this

quantity is one of the $S_{c,z,k}$ values considered in Equations 12 and 26 above, with the non-zero values only for z=k because this space put is non-transportable;
in software: ExchangeResults(Imports)

$SpcQty_{h,z}$ = physical quantity of space type h in zone z; determined in the SD Module and input to the AA Module (as the 'space quantities' information flow shown in Figure 1);
in software: FloorspaceI(Quantity)

$SpcPrp_{h,z}$ = proportion of physical quantity of space of type h in zone z that is available as a put and consumed in zone z;

$SpcBas_h$ = proportion of space for space type h available as a put when the unit exchange price for the space is at its reference level $PriceBas_h$;
in software: FloorspaceSupplyI(SupplyFunctionMidpointFactor);

$PriceBas_h$ = reference base price per unit for space type h for the proportion of space available as a put;
in software: FloorspaceSupplyI(SupplyFunctionMidpointPrice);

Δ_h = function coefficient for the rate of increase in the proportion of space available as a put for exponent term for space type h;
in software: FloorspaceSupplyI(SupplyFunctionDeltaFactor);

μ_h = function coefficient for the rate of increase in the proportion of space available as a put for linear term for space type h;
in software: FloorspaceSupplyI(SupplyFunctionSlopeFactor);

η_h = function coefficient for sensitivity to difference in exchange price for space type h concerning increase in the proportion of space available as a put for linear term for space type h;
in software: FloorspaceSupplyI(SupplyFunctionEta);

This is the same shape of function provided for import quantities. In one sense, the amount of space available as a put for consumption in a zone is analogous to the amount of a transportable put imported to the zone – hence the same functional form in both cases. In fact, the software treatment is the same, so if an import function for is specified for space (which is incorrect) it will be ignored in favour of the treatment here.

The coefficient Δ_h is positive and the coefficient μ_h is positive provided η_h is positive.

Total Demand and Total Supply Quantities

The quantity of each put c being bought in each exchange zone k by just the activities in the model– called the 'internal bought' quantity of put c in exchange zone k – is calculated

by summing the buying quantities from all zones allocated to the exchange zone, as follows:

$$TBInt_{c,k} = \Sigma_{z\varepsilon Z} B_{c,z,k} \quad (28)$$

where:

$TBInt_{c,k}$ = internal bought quantity of put c in exchange zone k, the total quantity of put c being bought in exchange zone k by all activities internal to the model area;
in software: ExchangeResults (InternalBought)

Similarly, the quantity of each put c being sold in each exchange zone k by just the activities in the model– called the 'internal sold' quantity of put c in exchange zone k – is calculated by summing the selling quantities from all zones allocated to the exchange zone, as follows:

$$TSInt_{c,k} = \Sigma_{z\varepsilon Z} S_{c,z,k} \quad (29)$$

where:

$TSInt_{c,k}$ = internal sold quantity of put c in exchange zone k, the total quantity of put c being sold in exchange zone k by all activities internal to the model area;
in software: ExchangeResults(InternalSold).

The aggregate demand for each put c in each exchange zone k is the total quantity of the put being bought in the exchange zone, including the internal bought quantity and the export quantity in the zone. This aggregate demand for each put c in exchange zone k is calculated as follows:

$$TDem_{c,k} = Qe_{c,k} + \Sigma_{z\varepsilon Z} B_{c,z,k} \quad (30)$$

where:

$TDem_{c,k}$ = aggregate demand for put c in exchange zone k by all activities in the model area;
in software: ExchangeResults(Demand);

Similarly, the aggregate supply for each put c in each exchange zone k is the total quantity of the put being sold in the exchange zone, including the internal sold quantity and the import quantity in the zone. This aggregate supply for each put c in exchange zone k is calculated as follows:

$$TSup_{c,k} = Qi_{c,k} + \Sigma_{z \varepsilon Z} S_{c,z,k} \quad (31)$$

where:

$TSup_{c,k}$ = aggregate supply for put c in exchange zone k by all activities in the model area;
in software: ExchangeResults(Supply).

The AA Module seeks an equilibrium solution where the aggregate demand equals the aggregate supply for each put for each exchange zone. This amount of equal supply and demand is called the 'exchange quantity' for the zone, where:

$$TE_{c,k} = TDem_{c,k} = TSup_{c,k} \quad (32)$$

where:

$TE_{c,k}$ = exchange quantity for put c in exchange zone k.

The AA Module seeks this equilibrium solution using an iterative process – as is described in detail in the next section. The progress towards this equilibrium solution is monitored (at least in part) using the residual between the aggregate demand and the aggregate supply for each put in each exchange zone, as follows:

$$Residual_{c,k} = TSup_{c,k} - TDem_{c,k} \quad (33)$$

where:

$Residual_{c,k}$ = the residual surplus quantity of supply over demand for put c in exchange zone k;
in software: ExchangeResults(Surplus).

At the theoretical equilibrium solution, this residual surplus is 0 in all cases – for all puts and all exchange zones. In practical work, as a compromise avoiding excessively long runtimes, the process of iteratively seeking the equilibrium solution may be terminated when the residual surplus values satisfy specified convergence criteria indicating they are sufficiently close to 0.

5.4. Size terms in the Activity Allocation Module

Size terms are an important component of random utility modelling. Aggregations of opportunities, such as LUZs, should be chosen in proportion to their size, all other things being equal. If a large zone is split into two smaller zones of equal size, the two zones together shouldn't generally attract more nor less choices than the larger combined zone. But, the measurements for size need to reflect the number of opportunities, and so different size metrics are required for different types of models. This section describes the size terms used at the different levels of the AA module.

The locations of production, consumption, and exchange in AA are LUZs. That is, the AA module allocates production, consumption, and exchange to LUZs. The random utility theory that AA is based on (see section 5.6. Derivation of Allocation Equations Using Random Utility Theory), starts from individual locations within each LUZ. The aggregation to an entire LUZ necessitates a size term, to remove the influence of zone boundaries, with the resulting attractiveness of an LUZ being the expected maximum of the individual alternatives (continuous or discrete) in the LUZ. (This is a standard approach in logit modelling, but careful treatment is required since, unlike most choice models, AA often has alternatives that differ in size by orders of magnitude for some purposes, and is a supply/demand model with endogenous prices, not simply a demand model.) Without size terms, small zones could end up with extreme prices — either very high prices to stop consumers from trying to consume too much, or very low prices to stop producers from trying to sell too much for the size of the zone. We can't control relative sizes by redrawing zone boundaries, since a zone can be large for some purpose but small for other

purposes (e.g. a large residential subdivision is small for job locations), or small in the current year (no development yet) but large in the future.

AA allocates activities to zones and to technology options in a nested logit model, and then the puts are allocated to (or from) exchange locations using logit models in the additive logit framework. Size measurements and associated size terms occur at these three different levels. We try to maintain the convention that the phrase "size" or "size measurement" refers to the appropriate measurement of size, S, while "size term" refers to the equivalent utility impact of size, generally $\frac{1}{\lambda}\ln(S)$.

For the lower level, of allocation of puts to exchange zones, a boostrapping procedure is used to allocate activities to zones based on floorspace or constraints. This bootstrapping procedure sets a flat price landscape, then allocates activities into zones and space using these flat prices, and their production/consumption in this state is used to measure the opportunities and set the size for the commodities. A column called TotalSizeTerm needs to be present in the CommoditiesI.csv file so that this bootstrapping procedure is working at the right order of magnitude, and a column called ConstInExchangeSize needs to be placed in the file ActivitiesI.csv, and set to TRUE, so that constraints (e.g. in the base year) influence these size terms. The values of the TotalSizeTerm column are usually set from the model wide amount used (consumption) of each commodity from the calibrated base-year MakeUse.csv file.

At the middle level, of technology choice, each technology option involves consumption of one type of space. The size terms for each technology option are the portion of that space type allocated in that zone.

At the top level, of LUZ home (residential or business) location choice for the activity, size terms are not needed, as the size terms from the middle level of technology flow directly up. (There is a facility in the file ActivitySizeTermsI to relate top-level size terms to quantities of floorspace, but this is not used when the consumption of space is explicit in the technology options, as is the case of the SWIM model.)

This direct and explicit calculation of size terms is embedded in the newer versons of the PECAS software.

Activity Benefit Measures

The model provides allocations of activity quantities by activity category by zone, put flow quantities from production zone to consumption zone via exchange zone, imports and exports by exchange zone and exchange prices by put by exchange zone – all consistent with the random utility maximization theory of choice behavior.

Also consistent with this random utility maximization theory, the model provides the basis for assessing the changes in model-wide consumer surplus by production activity. In particular, the model-wide location composite utility for a given activity category is determined consistent with Equations 01 and 02 as follows:

$$CLU_a = (1/\lambda_{l,a}) \cdot \ln \left(\Sigma_{z \in Z} \exp (\lambda_{l,a} \cdot LU_{a,z}) \right) \quad (34)$$

where:

CLU_a = model-wide location composite utility for a unit of activity a;
in software: ActivitySummary(CompositeUtility).

Differences in CLU_a between two model runs can be used to calculate measures of the changes in consumer benefits arising with the differences in policy actions for the two model runs. These measures can help assess quality of life and business attractiveness for different household and industrial categories as represented.

5.5. Solution Algorithm

The exchange zones simulate markets where the supply for each put, which is price elastic, meets the demand for the put, which is also price elastic with the opposite sign. The AA Module seeks the equilibrium solution for all these markets, adjusting the exchange prices in the exchange locations working towards the exact point where all the

markets clear, that is, where $TDem_{c,k} = TSup_{c,k}$ (and where $Residual_{c,k} = 0$) for all c and k.

The AA Module seeks this equilibrium solution in a series of iterations. In each such iteration, the allocations are made according to current prices, and the prices are then updated according to the supplies and demands arising with these allocations. A flow chart of the process is shown in Figure 5. The general flow of the process is counter-clockwise in the depiction in Figure 5. The process starts with an initial set of prices for the current prices. This initial set of prices can be either specified by the user as an input to the model or taken from the solution of a previous run of the process.

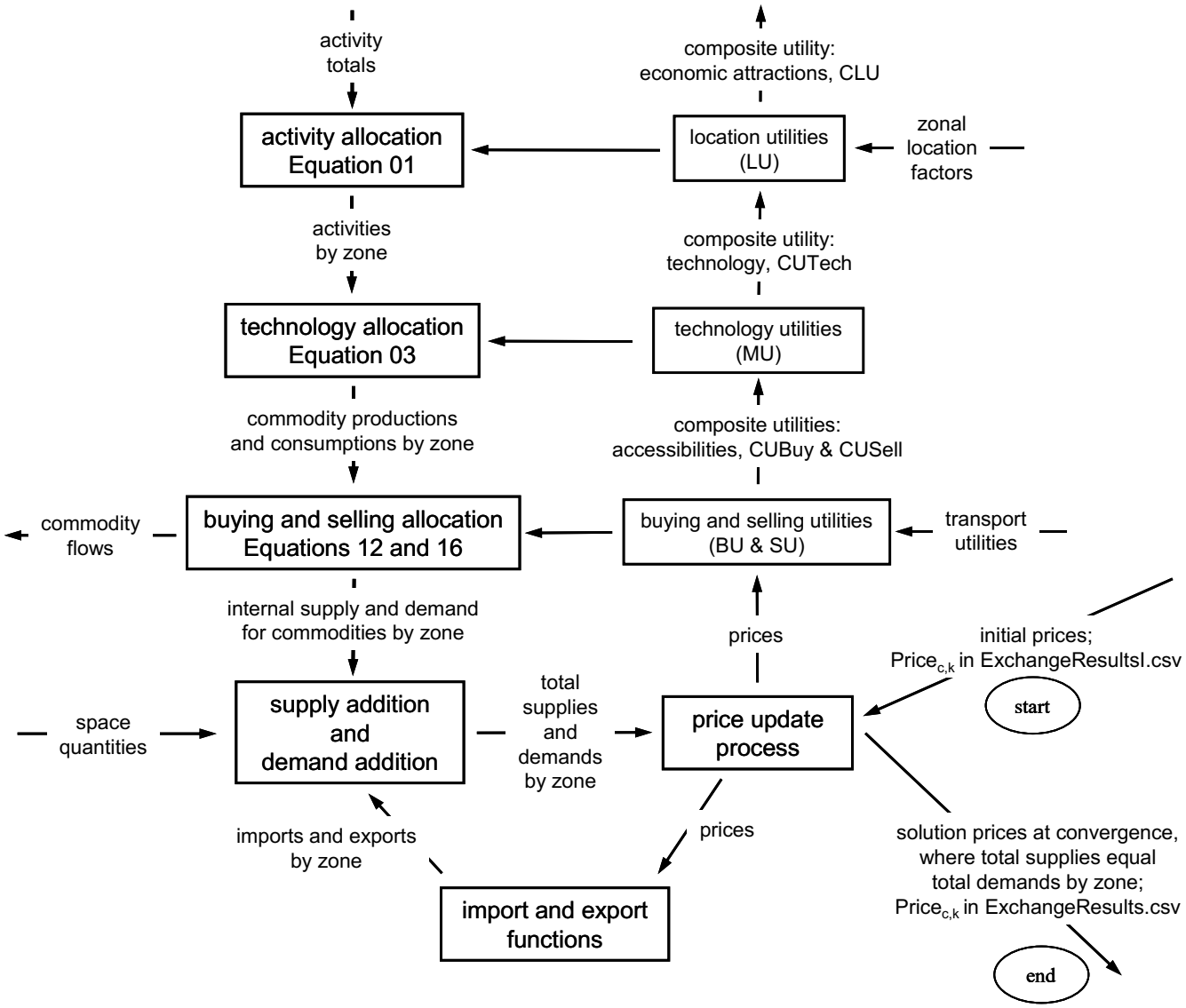

Figure 5: Allocation and price updating process used in search for equilibrium solution

The extent that the current prices do not satisfy the condition that $Residual_{c,k} = 0$ for all c and k is measured using a weighted sum-of-squares value as follows:

$$MClear = \Sigma_{c\varepsilon C} VergeWt_c^2 \Sigma_{k\varepsilon K} (Residual_{c,k})^2 \quad (35)$$

where:

MClear = residual-squared measure of the extent that all markets have not cleared and the condition that $Residual_{c,k} = 0$ for all c and k has not been satisfied; a value of 0 indicates the condition has been fully satisfied
in software: Merit Measure value reported in logfile

$VergeWt_c$ = weight applied to residual-squared values for put c to account for units effects when summing residual-squared values across puts;
in software: CommoditiesI(GOFWeighting).

The price update process – at the heart of the solution process – seeks to minimize MClear. In each iteration, it calculates the partial derivatives of the total excess demand (excess of demand by buyers and exporters over supply by sellers and importers) in each exchange zone with respect to the price in that exchange zone. The derivatives with respect to other prices in other zones or for other puts are assumed to be zero, and a price change in each zone is calculated. This price change is multiplied by the current step size.

In addition, the first derivative matrix of the total excess supply model-wide for every put with respect to the average (model-wide) price of every put is calculated. This derivative matrix is used to solve for the change in model-wide average prices that would result in a model-wide excess supply (surplus) of zero.

The Levenberg-Marquardt method is used to apply the current step size to this average price change calculation. The diagonal of the first derivative matrix is divided by the step size, which is typically less than 1.0 . Thus, the direction of the step is adjusted as the

step size changes: a smaller step size increases the diagonal elements relative to the other elements of the first derivative matrix, making the process more like a steepest descent algorithm and less like Newton's first-derivative algorithm.

The two price changes are added together – the average price change using the exact derivative matrix (which applies to every exchange), and the local price change using the partial derivative of local surplus with respect to local price. The local price change is multiplied by the Local Step Size Adjustment, StepLoc, which is specified in the inputs to the AA Module (*in software: pi.properties(pi.localPriceStepSizeAdjustment)*). The value of StepLoc can be used to adjust the relative contribution of average price change using an exact Newton's method and the local price change using a local derivative.

If a step results in a lower value for MClear, then the step adjustment factor is increased slightly for the next iteration. If a step results in a higher value for MClear, then the step is abandoned, the step adjustment factor is reduced substantially, and a new and smaller step is calculated to replace the abandoned one. The initial, maximum and minimum step sizes, StepInit, StepMax and StepMin, respectively, are specified in the inputs to the AA Module (*in software: pi.properties(pi.initialStepSize), pi.properties(pi.maximumStepSize) and pi.properties(pi.minimumStepSize) respectively*). If the current step size results in a numerical overflow, the step size is reduced even if doing so would result in it being below the minimum.

At the theoretical equilibrium solution, the value for MClear is 0, indicating the residual surplus, $Residual_{c,k}$, is 0 in all cases for all puts and all exchange zones, for all c and k. In practical work, as a compromise avoiding excessively long runtimes, the process of iteratively seeking the equilibrium solution is terminated when either (a) the residual surplus values satisfy specified convergence criteria indicating they are sufficiently close to 0, or (b) the specified maximum number of iterations is reached.

5.6. Derivation of Allocation Equations Using Random Utility Theory

The three levels of allocation equations used in the AA Module can be derived from a single utility function using random utility theory. This derivation is outlined here. For a more complete treatment, see (Abraham & Hunt, 2007).

Joint Choice

For one unit of activity type $a \in A$, where A consists of the full set of types of activity under consideration, including households, business establishments, and other institutions, consider the joint choice of:

- Location, $l \in L$, that is the home location for the unit; being residential location for households, or establishment location for business establishments and other institutions;

- Technology Option, $p \in P^a$, described by a set of technical coefficients $\alpha_p = \{\alpha_{p1}, \alpha_{p2}, ...\alpha_{pn}, ..., \alpha_{pN_p}\}$ and a corresponding list of puts $c_p = \{c_{p1}, c_{p2}, ..., c_{pn}, ..., c_{pN_p}\}$, each $c_{pn} \in C$. Each α_{pn} describes how much of put c_{pn} is produced (or consumed, if α_{pn} is negative) per unit of activity a, with indices n from 1 through N_p. P^a is the set of allowed Technology Option alternatives for activity a; and

- Exchange location, $e_n \in E_c$, for each put c_{pn} produced or consumed, being the choice of where to purchase, sell (or otherwise exchange as is the case for unpriced puts) the quantity $|\alpha_{pn}|$.

The utility of this joint choice is given by:

$$U^a_{lpe_1e_2\cdots e_n} = V^a_l + \varepsilon^a_l + V_p + \varepsilon_{lp} + \sum_{n=1\cdots N} |\alpha_{pn}| s_{pn} \left(V_{e_n l} + \varepsilon_{e_n lp}\right) \quad (36)$$

where:

V_l^a = the measurable component of utility associated with the location *l* and activity *a*

ε_l^a = a random component of utility associated with location *l* and activity *a*

V_p = the measurable component of utility associated with the technology option *p*

ε_{lp} = a random component of utility associated with the technology option *p* and location *l*

α_{pn} = the technical coefficients associated with technology option *p* as described above

s_{pn} = scaling adjusting associated with technical coefficient α_{pn} (non-negative and usually 1.0)

$V_{e_n l}$ = the measurable component of utility associated exchanging the put c_{pn} associated with α_{pn} in exchange location e_n given location *l* and technology option *p*

$\varepsilon_{e_n lp}$ = a random component of utility associated with exchanging the put c_{pn} at exchange location e_n given activity location *l* and technology option *p.*

Exchange Location Choice – Buying and Selling Allocations

Exchange location choice relates to the buying and selling allocations at the lowest level of the three-level allocation. It concerns the probability of choosing exchange location e_1 for put c_1 given the choice of location *l* and technology option *p*, which is the probability that one of the set of joint choice alternatives involving e_1 will have a higher utility than any of the set of joint choice alternatives not involving e_1

$$\Pr(e_1) \mid lp = \Pr\left(U_{lpe_1e_2\ldots e_n} \geq U_{lpe_1'e_2'\ldots e_n}\right) \forall e_1' \in E, e_1' \neq e_1 \tag{37}$$

A standard logit model treatment is used for the lower level of nested models: it is assumed that the $\varepsilon_{e_1 lp}$ and $\varepsilon_{e_1' lp}$ are independently and identically distributed with Gumbel distributions with location parameter 0 and scale parameter μ_{cb} if α_{pn}^a is negative (representing puts consumed) or μ_{cs} where α_{pn}^a is positive (representing puts produced).

It is also assumed that $V_{e_n l}$ takes one of two values, V_{ecls} if the location is *l,* the exchange location is *e*, the put c_{pn} is *c* and α_{pn} is positive, or V_{eclb} if the location is *l,* the exchange location is *e*, the put c_{pn} is *c* and α_{pn} is negative.

Then for put *c* sold from location *l* by any activity *a* under any technology option *p* the probability of selling to exchange *e* is:

$$\Pr(e)|\, cls = \frac{\exp(\mu_{cs} V_{ecls})}{\sum_{e' \in E_c} \exp(\mu_{cs} V_{ecls})} \tag{38}$$

and for put *c* bought to be consumed in location *l* by any activity *a* under any technology option *p* the probability of buying in exchange *e* is:

$$\Pr(e)|\, clb = \frac{\exp(\mu_{cb} V_{eclb})}{\sum_{e' \in E_c} \exp(\mu_{cb} V_{eclb})} \tag{39}$$

The formulae used for V_{ecld} and V_{ecls} take into account the price of the put at the exchange $\pi_{c,e}$, as well as the utility of transporting the put either from the production location to the exchange (for V_{ecls}) or the from the exchange to the consumption location (for V_{eclb}).

Technology option choice – Technology Allocations

Technology option choice relates to the technology allocations at the middle level of the three-level allocation. It concerns the probability of choosing technology option *p* given the choice of location *l*, which is the probability that one of the set of joint choice alternatives involving *p* will have a higher utility than any of the set of joint choice alternatives not involving *p*.

$$\Pr(p)|\,l = \Pr\left(V_l^a + \varepsilon_l^a + V_p + \varepsilon_{lp} + \max_{e_1 \ldots e_n}\left(\sum_{n=1\cdots N_p} \left|\alpha_{pn}\right| s_{pn}\left(V_{e_n l} + \varepsilon_{e_n lp}\right)\right) \geq \right.$$
$$\left. V_l^a + \varepsilon_l^a + V_{p'} + \varepsilon_{lp'} + \max_{e_1 \ldots e_n}\left(\sum_{n=1\cdots N_p} \left|\alpha_{p'n}\right| s_{p'n}\left(V_{e_n l} + \varepsilon_{e_n lp'}\right)\right)\right) \quad \forall p' \in P^a, p' \neq p \tag{40}$$

V_l^a and ε_l^a are associated with l, which is given, so are constant and can be dropped from the equation. The $V_{e_n l} + \varepsilon_{e_n lp}$ terms have been assumed to be independent of each other, so the maximum of their sum is the sum of their maximum. The Gumbel distribution is transformable linearly so the distribution of $\left|\alpha_{pn}\right| s_{pn}\left(V_{e_n l} + \varepsilon_{e_n lp}\right)$ is also Gumbel distributed.

The distribution of the maximum of a set of independently identically distributed Gumbel variables is also Gumbel.

Together these properties lead to the distribution of $\max_{e_n}\left(\left|\alpha_{pn}\right| s_{pn}\left(V_{e_n l} + \varepsilon_{e_n lp}\right)\right)$ being Gumbel distributed with location parameter equal to $\frac{\left|\alpha_{pn}\right| s_{pn}}{\mu_{cd}} \ln\left(\sum_{e_n} \exp\left(\mu_{cd} V_{e_n l}\right)\right)$ and scale parameter equal to $\frac{\mu_{cd}}{\left|\alpha_{pn}\right| s_{pn}}$ where

$$\mu_{cd} = \begin{cases} \mu_{cs} \text{ if } \alpha_{\text{pn}} \text{ is positive} \\ \mu_{cb} \text{ if } \alpha_{\text{pn}} \text{ is negative} \end{cases} \tag{41}$$

Let $\widetilde{V}_{cl} = \frac{1}{\mu_{cd}} \ln\left(\sum_{e_n} \exp\left(\mu_{cd} V_{e_n l}\right)\right)$ (42)

and

$$\varepsilon_{c_{pn}l} = \max_{e_n}\left(\left|\alpha_{pn}\right| s_{pn}\left(V_{e_n l} + \varepsilon_{e_n lp}\right)\right) - \left|\alpha_{pn}\right| s_{pn}\widetilde{V}_{cl} \quad (43)$$

Then $\varepsilon_{c_{pn}l}$ is Gumbel distributed with location parameter 0 and scale parameter $\frac{\mu_{cs}}{\left|\alpha_{pn}\right| s_{pn}}$ if

$\alpha_{pn}>0$ or $\frac{\mu_{cb}}{\left|\alpha_{pn}\right| s_{pn}}$ if $\alpha_{pn}<0$,

and:

$$\Pr(p)\,|\,l = \Pr\left(V_p + \varepsilon_{lp} + \sum_{n=1\ldots N_p}\left|\alpha_{pn}\right| s_{pn}\widetilde{V}_{cl} + \sum_{n=1\ldots N_p}\varepsilon_{c_{pn}l} \geq V_{p'} + \varepsilon_{lp'} + \sum_{n=1\ldots N_{p'}}\left|\alpha_{p'n}\right| s_{p'n}\widetilde{V}_{cl} + \sum_{n=1\ldots N_{p'}}\varepsilon_{c_{p'n}l}\right)$$
$$\forall p' \in P^a, p' \neq p$$

The intention is to have independent and identical distributions of ε_{lp} such that $\varepsilon_{lp} + \sum_{n=1\ldots N_p}\varepsilon_{c_{pn}l}$ for each technology option *p* are all independently and identically Gumbel distributed, as this will lead to a closed form function for the probability of choosing *p.* This sum represents the uncertainty associated with choosing exchange locations for each of the puts exchanged under technology option *p*, plus the uncertainty associated with technology option *p* itself. The terms are added together in this way because the unit of activity a (the business establishment or household) must choose exchange locations for each of the puts.

The sum of independent Gumbel distributions is complicated. Fortunately, it is not necessary here to determine the exact distribution of $\sum_{n=1\ldots N_p}\varepsilon_{c_{pn}l}$, only enough about it in order to be able to assign a reasonable Gumbel distribution to $\varepsilon_{lp} + \sum_{n=1\ldots N_p}\varepsilon_{c_{pn}l}$. By the

Central Limit Theorem, when N is large the distribution of $\sum_{n=1...N_p} \varepsilon_{c_{pn}l}$ approaches the Normal distribution with mean $\Gamma'[1] \sum_{n=1...N_p} \frac{1}{\mu_{c_n d} \left|\alpha_{pn}\right| s_{pn}}$ and variance $\frac{\pi^2}{6} \sum_{n=1...N_p} \left(\frac{\left|\alpha_{pn}\right| s_{pn}}{\mu_{c_n d}} \right)^2$ where $\Gamma'[1]$ = Euler's constant, 0.577216. This is because the mean of a Gumbel distribution with location *a* and scale μ is $a + \frac{\Gamma'[1]}{\mu}$, and the variance is $\frac{\pi^2}{6\mu^2}$; where $\Gamma'[1]$, Euler's constant, is the first derivative of the gamma function with respect to n at n=1. The number of puts in currently operation policy forecasting models, N, is between 30 and 100, which is approaching large in this context.

A distribution is assumed for ε_{lp} that adds to the variance of $\sum_{n=1...N_p} \varepsilon_{c_{pn}l}$ (it must increase the variance if it is independently distributed), shaped so that $\varepsilon_{lp} + \sum_{n=1...N_p} \varepsilon_{c_{pn}l}$ has a Gumbel distribution with variance > $\max_{p \in P^a} \left(\frac{\pi^2}{6} \sum_{n=1 \cdots N_p} \left(\frac{\left|\alpha_{pn}\right| s_{pn}}{\mu_{c_n d}} \right)^2 \right)$, identical for each *p*, giving scale parameter

$$\mu_a^p < \min_{p \in P^a} \left(\frac{1}{\sqrt{\sum_{n=1 \cdots N_p} \left(\frac{\left|\alpha_{pn}\right| s_{pn}}{\mu_{c_n}} \right)^2}} \right), \qquad (44)$$

and location parameter 0. Thus the probability of choosing technology option *p* given location *l* is

$$\Pr(p)\,|\,l = \frac{\exp\left(\mu_a^p\left(V_p + \sum_{n=1\cdots N_p}\left|\alpha_{pn}\right| s_{pn}\widetilde{V}_{cl}\right)\right)}{\sum_{p'\in P^a}\exp\left(\mu_a^p\left(V_{p'} + \sum_{n=1\cdots N_{p'}}\left|\alpha_{p'n}\right| s_{p'n}\widetilde{V}_{cl}\right)\right)} \tag{45}$$

$\widetilde{V}_{cl}$ is the composite utility, or expected maximum utility, of selling (if α_{pn} is positive) or buying (if α_{pn} is negative) put *c* from (or to) the available set of exchanges when located at l, taking into account price and transport. These are measures of the business conditions and living conditions at l, and the probability of choosing a particular technology option is a function of these conditions, as shown by Equation 10. These are analogous to accessibility measures calculated in transportation demand models, except that they also take into account prices for goods and services.

If the $V_{e_n l}$ (V_{ecls} and V_{eclb} for each putcommodity) are calculated using consistent price coefficients, then it is reasonable to assume that the s_{pn} are 1.0, meaning a dollar spent or earned on one putcommodity has the same utility as a dollar spent or earned on another.

Location Choice – Activity Allocation to Zones

Location choice relates to the activity allocations to zones at the highest level of the three-level allocation. It concerns the probability of choosing location *l*, which is the probability that one of the set of joint choice alternatives involving *l* will have a higher utility than any of the set of joint choice alternatives not involving *l*:

$$\Pr(l) = \Pr\left(V_l^a + \varepsilon_l^a + \max_{pe_1\ldots e_n}\left(V_p + \varepsilon_{lp} + \sum_{n=1\cdots N_p}\left|\alpha_{pn}\right| s_{pn}\left(V_{e_n l} + \varepsilon_{e_n lp}\right)\right) \geq\right.$$

$$V_{l'}^{a} + \varepsilon_{l'}^{a} + \max_{pe_1 \ldots e_n} \left(V_p + \varepsilon_{lp} + \sum_{n=1 \cdots N_p} \left| \alpha_{p'n} \right| s_{p'n} \left(V_{e_n l} + \varepsilon_{e_n l p'} \right) \right) \quad \forall l' \in L, l' \neq l \tag{46}$$

Using nested logit theory with the assumptions indicated above regarding the distribution of ε_{lp} and further assuming that ε_l^a has a shape such that the sum $\varepsilon_l^a + \max_p \left(\varepsilon_{lp} + \sum_{n=1 \ldots N_p} \varepsilon_{c_{pn} l} \right)$ has a Gumbel distribution, the result is

$$\Pr(l) = \frac{\exp\left(\mu_a^l \left(V_l^a + \widetilde{V}_{la} \right)\right)}{\sum_{l' \in L} \exp\left(\mu_a^l \left(V_{l'}^a + \widetilde{V}_{l'a} \right)\right)} \tag{47}$$

with $$\mu_a^l < \mu_a^p \tag{48}$$

and $$\widetilde{V}_{la} = \frac{1}{\mu_a^p} \ln \left(\sum_{p \in P^a} \exp \left(\mu_a^p \left(V_p + \sum_{n=1 \ldots N_p} \left| \alpha_{pn} \right| s_{pn} \widetilde{V}_{cl} \right) \right) \right) \tag{49}$$

Thus the probability of a unit of activity *a* choosing a location *l* is largely a function of the different accessibility measures $\widetilde{V}_{cl}$, weighted according to the technical coefficients α_{pn} for the technology option. This is an advantage when the $\widetilde{V}_{cl}$ are consistent and the s_{pn} are set to 1.0, because it overcomes the severe difficulties that can arise in the estimation of the relative impacts of different highly correlated accessibility measures from location choice observations. The s_{pn} factors can be changed from 1.0 if the $\widetilde{V}_{cl}$ are not specified consistently, or if observations of location choice behaviour provide more precise guidance

Benefit Measures

The expected maximum utility of the full joint choice for a unit of activity type *a* is then given by:

$$\frac{1}{\mu_a^l}\sum_{l\in L}\exp\left(\mu_a^l\left(V_l^a+\tilde{V}_{la}\right)\right) \tag{50}$$

The difference in this value between model runs considering different cases provides a measure of the change in consumer surplus or producer surplus of choosing amongst the full set of options provided by the land use/transport system regarding location, technology option, and exchange location in the two cases. This is a key output for policy analysis.

6. Space Development Module – Disaggregate Version

6.1. Approach

The Space Development Module (SD Module) uses a disaggregate approach. It works through a list of the cells or parcels of land in each LUZ, considering each cell or parcel one after another, in each year-to-year step that the module is run.

Each cell or parcel (called 'parcel') of land has a set of attributes, including among other items:

- A quantity of existing developed space (called 'space') of one type with a specific age;
- A set of zoning rules specifying the types of space that are permitted and the densities at which they are permitted;
- The costs and fees associated with development of each permitted space type and quantity; and
- The price (rent) for each type of permitted space.

Quantities of space on parcels are measured in areas, such as square feet or square meters. Space of a given type is a putcommodity consumed by activities in the AA Module. Space is 'non-transportable' in that it must be consumed in the zone where it is located. The aggregate quantity of space of a given type in a given zone is the sum of the quantities on all of the parcels in the zone.

Developers act to change the types of space and/or the quantities of space on parcels. The consideration of the cell or parcel in a given year-to-year step includes determining the development event for the year-to-year step, establishing whether or not the existing space is changed by some form of developer actions during the step, and, if the space is changed, what the updated space type and quantity are to be. Keeping the space the

same in a given year-to-year step is also a development event in this context, as is allowing the space to go derelict through neglect.

In a given year-to-year step, logit models are used to assign selection probabilities to each of the events that are permitted for the parcel according to the zoning rules, including each of the permitted updated space types and quantities options, along with the 'no change' and 'derelict' events. The utility functions in these logit models calculate the expected net revenues to the developer for the available options, incorporating the prices and the costs for transition, maintenance and servicing in each case.

If the Monte-Carlo implementation of SD is chosen, one of these event is then selected using a random number generator and results recorded, and the process moves on to the next parcel. If the deterministic microsimulation version of SD is chosen (Hunt et al., 2019), the probabilities are used to calculate the sum of the expected quantities for each LUZ, and then these amounts are allocated to the best parcels in each LUZ using a matchmaker algorithm.

The space prices that are considered are expressed per unit of space per unit of time considered in the AA Module, and thus are rents for the use of the space for a given time and not prices for purchase of the space or the associated land for ownership in perpetuity.

6.2. Typical Categories in Implementation

A typical set of space types for an implementation of the SD Module is as follows:

- Low density housing
- Medium density housing
- High density housing
- Industrial space

- Warehouse space
- Retail space
- Office Grade A space
- Office Grade B space
- Hospital space
- Gradeschool space
- Post-Secondary Institutional space
- Government space
- Agricultural space

These must relate to the space types included as individual put categories in the AA Module, although the relationship between space types in SD and floorspace put categories in AA can be many-to-many. In general, standard floorspace categories in land use descriptions are used as the starting point for defining the set of space types, but, for example there could be an “office/retail” space type in SD that is 60% office space in AA and 40% retail space in AA, and used to represent mixed-use buildings.

A derelict version of each space type is also included. The ‘derelict’ development event is the transition to the derelict version of the space type, usually arising because of developer inaction that is possible in any case and likely more probable when the rents for all allowable space types are low. With a derelict version of each space type included, the SD Module can take into account the differences between the costs for re-development on land with derelict space and the costs for new development on raw land.

A parcel need not contain any developed space. In fact, many parcels will contain no developed space at one or more points in time. Such parcels are designated 'empty'. This 'empty' designation is one of the development states available in all models generally, with separate representation of the influences on transitions into and out of the empty state included.

6.3. Mathematical Formulation

Consideration of Development Possibilities

In a given year-to-year step, going from the previous year to the next year, the price for each type of space in each LUZ determined by the AA Module for the previous year is input to the SD Module. This 'space prices' information flow is shown in Figure 1. The SD Module then starts the process of working through the list of parcels considering each parcel in turn. If the Monte-Carlo treatment is turned on, SD selects a development event and determins the updated (year t+1) conditions for the space on the parcel given the existing (year t) conditions and prices. If the deterministic algorithm is turned on, SD calculates the probability of each possible event and multiplies it by the expected value of the quantity amount given the event, to add to the expected value total for the LUZ. The steps in this process and the equations used in these steps are described below.

Skipping Parcels Both Undeveloped and Development Restricted

The zoning rules for the parcel are checked to see if any sort of developer action is permitted. If no developer action is permitted, as is the case for protected lands and forests for example, and there is no existing space on the parcel, then the process leaves the parcel as is and moves on to consider the next parcel.

Selecting updated space type and space quantity for parcel

Table 2 shows the range of possible development events for a parcel in a given year-to-year step in general.

Table 2: Set of possible development events

Event Name	Event Designation	Description
new space type and quantity	En	demolish any existing space and develop new space type and new space quantity as permitted by zoning rules
no change	E0	make no change, leaving the space type and space quantity the same
add	Ea	space type is kept the same, but the space quantity is increased within the range permitted by zoning rules and the age is reset to reflect the proportions of existing and new space
renovate	Er	space type is kept the same, but the age of the space is reset to 0
demolish	Ex	demolish any existing space and leave land empty
derelict	Ed	allow the space to become derelict

All of the development events listed in Table 2 are treated as 'transitions' in the year-to-year step, going from the existing space to the updated space on the parcel. The treatment of each of these categories of event is described further below. The action of completely demolishing a building and then constructing a new one of the same type is included in the first category of event, En.

The logit models representing the behavior of developers in the selection of a development event are linked in a nested logit structure consistent with the expected degrees of similarity among the alternatives and hence among the corresponding utility error terms. This structure is shown in Figure 6.

The selection of a development event and the resulting updated space type and updated space quantity for a parcel are done in a two-stage process. The first stage is selecting the development event and updated space type. The second stage is selecting the updated space quantity, if appropriate, conditional on the development event selected.

First Stage: Selecting Development Event and Updated Space Type

A nested logit model is used to assign selection probabilities to each of the development event alternatives available to the developer, then one of the alternatives is selected at random according to the distribution of these probabilities. The logit model utility value for each development event and corresponding space type option is the utility per unit of land, including the expected net revenues, for the developer in each case.

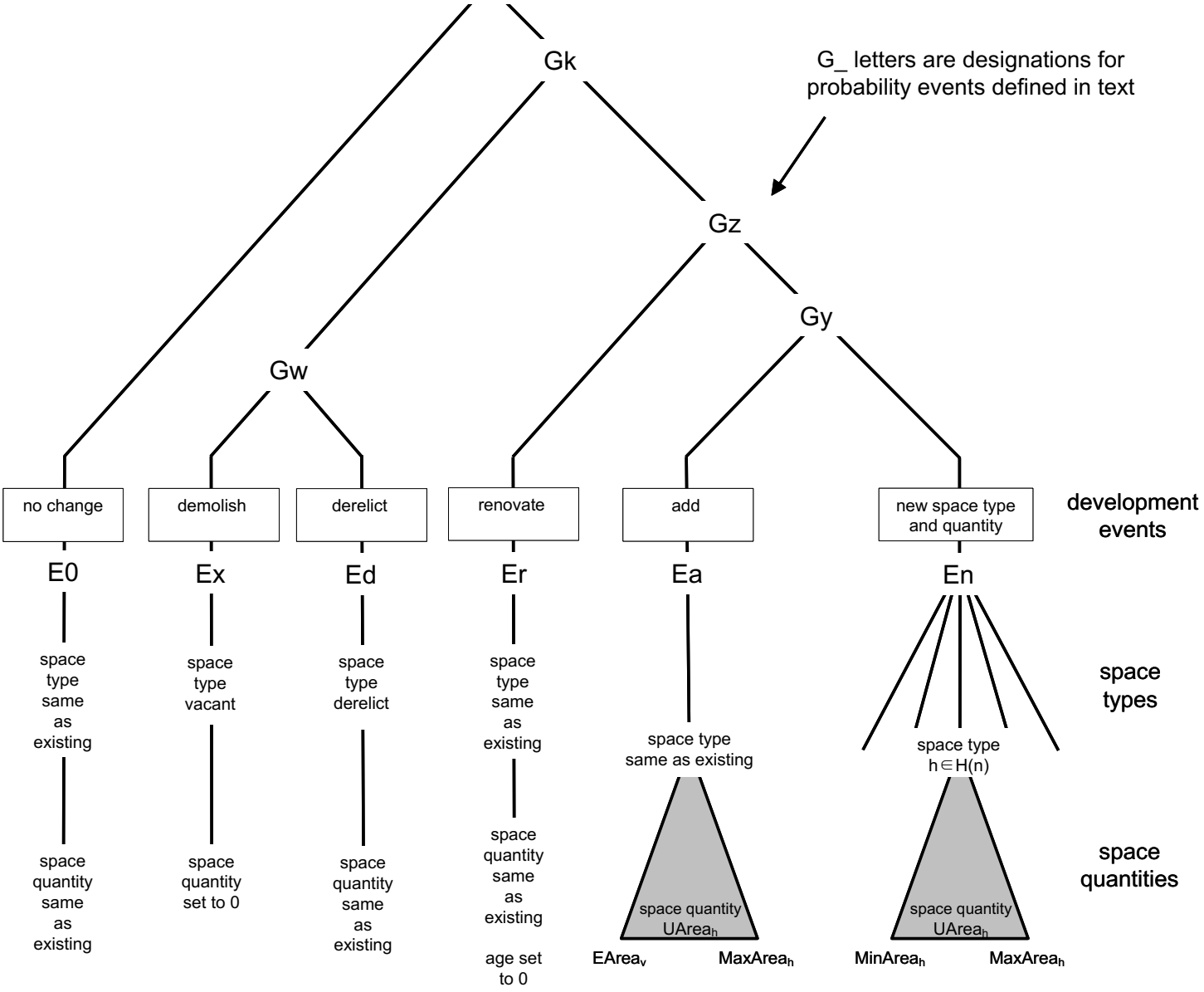

Figure 6: Nested logit model structure for developer event probabilities

'New Space Type and Quantity' Development Event (En)

One of the possible development events is 'new space type and quantity', designated En, where any existing space on the parcel is demolished and new space is developed. The updated space type h can be any of the types permitted by the zoning rules, including the same type as the existing space if permitted by the zoning rules, and the updated space quantity can be any of the quantities of the range permitted by the zoning rules. The utility per unit of land for this development event varies according to both the existing space type v and the updated space type h. For a given parcel, with an existing space type v, it is calculated for each possible updated space type h as follows:

$$RU_{v,h} = (1/(\lambda_{q,h} \cdot LotSize)) \cdot \{ \ln[IntL_h \cdot LotSize/(\lambda_{q,h} \cdot NetRev_{v,h})] - \ln[MaxArea_h - MinArea_h] \} + TrConst_{v,En} + TrConst_{v,h} \quad (51)$$

with:

$$IntL_h = \exp[\lambda_{q,h} \cdot ((NetRev_{v,h} \cdot MaxArea_h)/LotSize - TrCostsL_{v,h})] - \exp[\lambda_{q,h} \cdot ((NetRev_{v,h} \cdot MinArea_h)/LotSize - TrCostsL_{v,h})] \quad (52)$$

and;

$$NetRev_{v,h} = Rent_h - TrCostsS_{v,h} - MtCostsS_h \quad (53)$$

where:

v = index representing existing space type for parcel;
in software: stored for each parcel in parcel database, specified for base year in base year parcel database and then re-determined by SD in transitions each year after base year

h = index representing possible updated space type for parcel;
in software: SpaceTypesI(SpaceTypeID);

$RU_{v,h}$ = utility per unit of land for transitioning from existing space type v to updated space type h;

$MinArea_h$ = minimum allowable quantity of possible updated space type h for parcel;
in software: specified for each parcel in parcel database, built up from parcel level information on zoning rules

$MaxArea_h$ = maximum allowable quantity of possible updated space type h for parcel;
in software: specified for each parcel in parcel database, built up from parcel level information on zoning rules

LotSize = area of land on parcel; in units of land area
in software: specified for each parcel in parcel database

$IntL_h$ = value of integral term spanning range of area quantities from $MinArea_h$ to $MaxArea_h$ for space type h;

$NetRev_{v,h}$ = expected net revenue for transitioning from existing space type v to updated space type h on parcel per unit of updated space type h; in units of money per unit of area for space type h per SD time unit;

$Rent_h$ = expected rent for updated space type h on parcel per unit of updated space type h, constant within range of quantities from $MinArea_h$ to $MaxArea_h$; in units of money per unit of area for space type h per year;

$TrCostsS_{v,h}$ = amortized money cost for transitioning from existing space type v to updated space type h on parcel per unit of updated space type h, constant within range of quantities from $MinArea_h$ to $MaxArea_h$ (and typically this is the amortized construction unit cost for updated space type h); in units of money per unit of area for space type h per year;

$MtCostsS_h$ = maintenance money costs for updated space type h on parcel per unit of updated space type h, constant within range of quantities from $MinArea_h$ to $MaxArea_h$; in units of money per unit of area for space type h per year;

$TrCostsL_{v,h}$ = amortized money cost for transitioning from existing space type v to updated space type h on parcel per unit of land on parcel, constant within range of quantities from $MinArea_h$ to $MaxArea_h$; in units of money per unit of land per year;

$\lambda_{q,h}$ = utility function dispersion parameter for quantity allocation for updated space type h for new space development;
in software: SpaceTypes(QuantityChoiceDispersionParameter);

$TrConst_{v,En}$ = utility function constant for the 'new space type and quantity' development event with existing space type v on parcel, where the resulting 'transition' is from existing space type v to any possible new space type h
in software: SpaceTypesI(NewConstant);

$TrConst_{v,h}$ = utility function constant for transition from existing space type v to specific new space type h
in software: TransitionConstantsI(the v to h entry);

Equation 51 calculates the utility per unit of land for transitioning to updated space type h. It takes into account the full range of possible quantities of updated space type h permitted by the zoning rules, from $MinArea_h$ to $MaxArea_h$. This requires the calculation of a composite utility for a continuous range of lower-level alternatives within a nested logit model framework, which leads to the complexity of Equation 51.

The theory of the developer choice model in Equation 51 assumes piecewise constant values for each of the unit rates $TrCostsL_{v,h}$, $TrCostsS_{v,h}$, $MtCostsS_h$ and $Rent_h$ across the range of quantities from $MinArea_h$ to $MaxArea_h$. For practical implementation, if any of these unit rates cannot be represented appropriately using a single constant value across the full range of quantities from $MinArea_h$ to $MaxArea_h$, then the space type h should be split into separate categories where each of the unit rates can be represented appropriately using a single constant value.

Two constants are included, $TrConst_{v,En}$ and $TrConst_{v,h}$. This is to facilitate calibration of both the overall rate of transition to new space generally (using $TrConst_{v,En}$) and also the rates of transition to each specific new space type (using $TrConst_{v,h}$).

The determination of values for rent, $Rent_h$ and maintenance costs, $MtCostsS_h$, are described in subsequent sections below.

Included here is the case where the existing space is demolished and new space of the same type is constructed, in which case the transition constant $TrConst_{v,h}$ in Equation 51 is $TrConst_{v,h=v}$.

Also included here is the case where the building is modified substantially in order to accommodate a new activity. An example of this would be when a large house is converted into a small professional office. Even though there is no demolition per se, there is still a change in space type that in general involves substantial costs.

Not included here are the development events where there is no change to the space or the space is renovated rather than demolished and re-built. Those two development events are included below, with modified forms of the utility function components in Equations 51 through 53.

'No Change' Development Event (E0)

One of the possible development events is 'no change', designated E0, where the space is retained as is – with the same type and same quantity. In this case, the updated space type h is the same as the existing space type v. The updated space quantity is the same as the existing space quantity. There are no amortized money costs for transitioning. The only change is that the age of the space increases by one year, which in general will impact the rent and the maintenance costs as discussed in a subsequent section below.

The utility per unit of land for this development event, $RU_{v,E0}$, is calculated as follows:

$$RU_{v,E0} = (1/LotSize) \cdot (EArea_v \cdot NetRev_{v,E0}) + TrConst_{v,E0} \quad (54)$$

with:

$$NetRev_{v,E0} = Rent_h - MtCostsS_h \quad (55)$$

where:

$RU_{v,E0}$ = utility per unit of land for the 'no change' development event, where the updated space type is the same as the existing space type v and the updated quantity of space is the same as the existing quantity of space;

$EArea_v$ = quantity of existing space type v on parcel
in software: stored for each parcel in parcel database, specified for base year in base year parcel database and then re-determined by SD in transitions each year after base year

$NetRev_{v,E0}$ = expected net revenue associated with not changing existing space type v on parcel per unit of updated space type h (same as v); in units of money per unit of area for space type h per SD time unit;

$MtCostsS_h$ = maintenance money costs for updated space type h (same as v) on parcel; in units of money per unit of area for space type h per year;

$TrConst_{v,E0}$ = utility function constant for the 'no change' development event with existing space type v on parcel, where the resulting 'transition' is from existing space type v to updated space type h (same as v) that is one year older
in software: SpaceTypesI(NoChangeConstant);

Equations 54 and 55 above form a 'collapsed' version of Equations 51 through 53, where the range of allowable space quantities is restricted to just the existing quantity of space, there are no unit costs for transitioning, the rent and maintenance costs are for the existing space (getting 1 year older) and a separate 'no change' transition constant, $TrConst_{v,E0}$, is used in place of the transition constants $TrConst_{v,En}$ and $TrConst_{v,h}$.

In this 'no change' development event considered here, the 'transition' that is performed is the updated quantity and type of space are set to be the same as the existing quantity and type of space.

Not included here is the specific event where the space is demolished and re-built and the updated space type remains the same as the existing space type. That specific event is included as one of the possible events in the new space type and new space quantity options, at least to the extent that it is permitted by the zoning rules for the parcel. Also not included here is the specific event where the space is renovated. That specific event is included as one of the special development events considered below.

Typically, the 'no change' development event is the option selected in most cases – space tends to stay the same type and quantity from one year to the next – so RU_{E0} will in most cases have the highest of the utility values for the range of possible events and the resulting highest probability value.

'Add' Development Event (Ea)

One of the possible development events is 'add', designated Ea, where the type of space is kept the same but the quantity of space is increased by adding some new space, and the age of the space is reset to an average value that reflects the proportion of space that is new. Only increases in the quantity of space are considered.

In this case, the updated space type h is the same as the existing space type v. The updated space quantity is not known *a priori* and is handled in much the same way it is handled with the new space quantity events; that is, a composite utility for a continuous range of lower-level alternatives for the additional quantity of space is included in the utility for the space type and the quantity of additional space is selected in the second stage of the process. The utility per unit of land for the event is calculated taking into account the contributions of both (a) the utility arising with the existing space and (b) the utility arising with new space as follows:

$$RU_{v,Ea} = (1/LotSize) \cdot \{ (EArea_v \cdot NetRev_{v,E0}) + (1/\lambda_{q,h}) \{ \ln[IntL_{h=vEa} \cdot LotSize/(\lambda_{q,h} \cdot NetRev_{v,Ea})] - \ln[MaxArea_h - EArea_v] \} \} + TrConst_{v,Ea} \quad (56)$$

with:

$$IntL_{h=vEa} = \exp[\lambda_{q,h} \cdot ((NetRev_{v,Ea} \cdot MaxArea_h)/LotSize - TrCostsL_{v,h=vEa})] - \exp[\lambda_{q,h} \cdot ((NetRev_{v,Ea} \cdot EArea_h)/LotSize - TrCostsL_{v,h=vEa})] \quad (57)$$

and;

$$NetRev_{v,Ea} = Rent_h - TrCostsS_{v,h=vEa} - MtCostsS_h \quad (58)$$

where:

$RU_{v,Ea}$ = utility per unit of land for the 'add' development event, where the updated space type is the same as the existing space type v and the updated quantity of space is greater than the existing quantity of space reflecting the amount of space added;

$IntL_{h=vEa}$ = value of integral term spanning range of area quantities from $EArea_h$ to $MaxArea_h$ for space type h;

$NetRev_{v,Ea}$ = expected net revenue associated with the additional amount of space added to the existing space type v (becoming updated space type h) on parcel per unit of updated space type h; in units of money per unit of area for space type h per SD time unit;

$TrCostsS_{v,h=vEa}$ = amortized money cost for adding more of existing space type v (becoming updated space type h) on parcel per unit of updated space type

h, constant within range of quantities from $EArea_v$ to $MaxArea_h$ (and typically this is the amortized construction unit cost for updated space type h); in units of money per unit of area for space type h per year;

$TrCostsL_{v,h=vEa}$ = amortized money cost for adding more of existing space type v (becoming updated space type h) on parcel per unit of land on parcel, constant within range of quantities from $EArea_h$ to $MaxArea_h$; in units of money per SD land unit per year;

$TrConst_{v,Ea}$ = utility function constant for the 'add' development event with existing space type v
in software: SpaceTypesI(AddConstant);

Equations 56 through 58 above are an extension of Equations 51 through 53, where the utility for the existing space is combined with the expected utility for the additional space taking into account the range of possible amounts for the additional space, but only for one updated space type h that is the same as existing space type v. A separate 'add' constant, $TrConst_{v,Ea}$, is included in place of the transition constants $TrConst_{v,En}$ and $TrConst_{v,h}$.

The rent and maintenance costs for the existing space component, contributing to the expected net revenue associated with this component, $NetRev_{v,E0}$, are those for space type v (becoming space type h) for an age that is one year more than the age for the existing space. The rent and maintenance costs for the additional space component, contributing to $NetRev_{v,Ea}$, are those for space type h for an age of 0. The determination of these rent and maintenance cost values, for the two age amounts, is described in a subsequent section below.

The calculation of the new value for the age of the space requires the quantity of additional space being added. The quantity of additional space is selected in the second stage of the process, so the new value for the age of the space is also determined in the second stage of the process.

'Renovate' Development Event (Er)

One of the possible development events is 'renovate', designated Er, where the type of space is kept the same and the quantity of space kept the same, and the age of the space is reset to 0.

The utility per unit of land for this development event, $RU_{v,Er}$, is calculated as follows:

$$RU_{v,Er} = (1/LotSize) \cdot (EArea_v \cdot NetRev_{v,Er} - LotSize \cdot TrCostsL_{v,h=vEr}) + TrConst_{v,Er} \quad (59)$$

with:

$$NetRev_{v,Er} = Rent_h - TrCostsS_{v,h=vEr} - MtCostsS_h \quad (60)$$

where:

$RU_{v,Er}$ = utility per unit of land for the 'renovate' development event, where the updated space type is the same as the existing space type v and the updated quantity of space is the same as the existing quantity of space;

$NetRev_{v,Er}$ = expected net revenue associated with not changing existing space type v (becoming updated space type h for notation purposes only) on parcel per unit of updated space type h; in units of money per unit of area for space type h per SD time unit;

$TrCostsS_{v,h=vEr}$ = amortized money cost for renovating existing space type v (becoming updated space type h) on parcel per unit of updated space type h; in units of money per unit of area for space type h per year;
in software: DevelopmentTypesI(RenovationCost);
in new PA-database version of software: this is specified at the parcel level, built-up from parcel level information on fees, construction costs and zoning conditions penalty;

$TrCostsL_{v,h=vEr}$ = amortized money cost for renovating existing space type v (becoming updated space type h) on parcel per unit of land on parcel; in units of money per SD land unit per year;

$TrConst_{v,Er}$ = utility function constant for the 'renovate' development event with existing space type v on parcel, where the resulting 'transition' is from existing space type v to updated space type h that is the same as v and has an age of 0
in software: SpaceTypesI(RenovateConstant);

Equations 59 and 60 together form another 'collapsed' version of Equations 51 through 53, where the range of allowable space quantities is restricted to just the existing quantity of space, the costs per unit space and per unit land for transitioning are those for renovation, and the rent and maintenance costs (included in the expected net revenue) are those for renovated space that has an age of 0. A separate 'renovation' constant, $TrConst_{v,Er}$, is also included.

Not included here is the specific event where the space is demolished and re-built (not renovated) and the updated space type remains the same as the existing space type. That specific event is included as one of the possible events in the new space type and new space quantity options.

'Demolish' Development Event (Ex)

One of the possible development events is 'demolish', designated Ex, where the parcel becomes empty and the quantity of space is set to 0.

The utility per unit of land for this development event, $RU_{v,Ex}$, is calculated as follows:

$$RU_{v,Ex} = (1/LotSize) \cdot (- EArea_v \cdot TrCostsS_{v,Ex} - LotSize \cdot TrCostsL_{v,Ex}) + TrConst_{v,Ex} \quad (61)$$

where:

$RU_{v,Ex}$ = utility per unit of land for the 'demolish' development event, where the existing space type is v and updated space type is empty (actually the parcel is designated empty) and the updated space quantity is 0;

$TrCostsS_{v,Ex}$ = amortized money cost for demolishing existing space type v (and thereby removing all space) on parcel per unit of existing space type v; in units of money per unit of area for space type v per year; note that this cost rate is expressed per unit of the existing space type (that is being demolished) not the updated space type, which is different from most of the other cost rates parameters
in software: DevelopmentTypesI(DemolitionCost);

in new PA-database version of software: this is specified at the parcel level, built-up from parcel level information on fees, construction costs and zoning conditions penalty;

$TrCostsL_{v,Ex}$ = amortized money cost for demolishing existing space type v (and thereby removing all space) on parcel per unit of land on parcel; in units of money per SD land unit per year;
in software: ZoningSchemesI(LandFee);
in new PA-database version of software: this is specified at the parcel level, built-up from parcel level information on fees, construction costs and zoning conditions penalty;

$TrConst_{v,Ex}$ = utility function constant for the 'demolish' development event with existing space type v on parcel, where the resulting 'transition' is from existing space type v to no space (just an empty parcel)
in software: SpaceTypesI(DemolishConstant);

Equation 61 is another 'collapsed' version of Equations 51 through 53. In this case, the costs per unit space and per unit land for transitioning are those for demolition. The demolition cost per unit space relates solely to the existing space, so these transition costs are per unit of existing space. This is different from the other development events, where the transition costs per unit space are per unit of updated space. There is no rent or maintenance costs after the transition, so both $Rent_h$ and $MtCostsS_v$ are 0 and the net revenue term includes just the $TrCostsS_{v,Ex}$ term representing the demolition costs per unit of existing space. A separate 'demolition' constant, $TrConst_{v,Ex}$, is also included.

'Derelict' Developer Event (Ed)

One of the possible development events is 'derelict', designated Ed, where the space is allowed to become derelict. This is not to say that a conscious decision is made by a developer to transform the space into dereliction; rather, it is merely acknowledging that space sometimes does become derelict and, thus, this needs to be included as a possibility. The updated space quantity is the same as the existing space quantity, with the amortized money costs for transitioning and also the $NetRev_{v,h}$ in this case set to 0 and the utility per unit of land, $RU_{v,Ed}$, calculated using just the transition constant that remains as follows:

$$RU_{v,Ed} = TrConst_{v,Ed} \quad (62)$$

where:

$RU_{v,Ed}$ = utility per unit of land for the 'derelict' development event, where the updated space type is the 'derelict' form of the existing space type v and the updated quantity of space is the same as the existing quantity of space;

$TrConst_{v,Ed}$ = utility function constant for the 'derelict' development event with existing space type v on parcel;
in software: SpaceTypesI(DerelictConstant);

When the space on a parcel is transitioned to derelict, the space becomes the 'derelict' version of the space type it was before the transition. Thus, for each space type there is a corresponding derelict version of the space type, with its own set of values for the utilities for transitions representing further development. Only the 'new space type and quantity', 'no change', 'renovate' and 'demolish' development events are possible when the existing space is derelict. The other development events – 'add' and 'derelict' – do not apply.

When considering the transitions from derelict versions of space, additional constants are added to the utility values for these possible development events for existing derelict space, reflecting the differing costs structures applying with such transitions, as follows:

For 'new space type and quantity':

$$DRU_{v,h} = RU_{v,h} + DTrConst_{v,h=En} + DTrConst_{v,h} \quad (63)$$

where:

$DRU_{v,h}$ = utility per unit of land for transitioning from the derelict version of existing space type v to updated space type h;

$DTrConst_{v,En}$ = utility function constant for the 'new space type and quantity' development event with the derelict version of existing space type v on parcel, where the resulting transition is from existing space type v to any possible new space type h
in software: DerelictSpaceI(NewConstant);

$DTrConst_{v,h}$ = utility function constant for transition from the derelict version of existing space type v to updated space type h
in software: DerelictTransitionsConstantsI(the v to h entry);

Note that two constants are included in this case, $DTrConst_{v,En}$ and $DTrConst_{v,h}$, again, to facilitate calibration of both the overall rate of transition to new space generally (using $DTrConst_{v,En}$) and also the rates of transition to each specific new space type (using $DTrConst_{v,h}$).

For 'no change':

$$DRU_{v,E0} = RU_{v,E0} + DTrConst_{v,E0} \quad (64)$$

where:

$DRU_{v,E0}$ = utility per unit of land for transitioning from the derelict version of existing space type v to updated space type h;
$DTrConst_{v,E0}$ = utility function constant for the 'no change' development event with existing derelict version of space type v on parcel, where the resulting 'transition' is from the existing derelict space type v to the updated derelict space type h that is the same as v and one year older
in software: DerelictSpaceI(NoChangeConstant);

For 'renovate':

$$DRU_{v,Er} = RU_{v,Er} + DTrConst_{v,Er} \quad (65)$$

where:

$DRU_{v,Er}$ = utility per unit of land for the 'renovate' development event transitioning from the derelict version of existing space type v, where the updated space type is the same as the existing space type v and the updated quantity of space is the same as the existing quantity of space;
$DTrConst_{v,Er}$ = utility function constant for the 'renovate' development event with existing derelict version of space type v on parcel, where the resulting 'transition' is from existing space type v to updated space type h that is the same as v and has an age of 0
in software: DerelictSpaceI(RenovateConstant);

For 'demolish':

$$DRU_{v,Ex} = RU_{v,Ex} + DTrConst_{v,Ex} \quad (66)$$

where:

$DRU_{v,Ex}$ = utility per unit of land for the 'demolish' development event transitioning from the derelict version of the existing space type v, where the existing space type is v and updated space type is empty (actually the parcel is designated empty) and the updated space quantity is 0;

$DTrConst_{v,Ex}$= utility function constant for the 'demolish' development event with existing derelict version of space type v on parcel, where the resulting transition is from existing space type v to no space (just an empty parcel)
in software: DerelictSpaceI(DemolishConstant);

To calculate these utility values for transitions from derelict versions of existing space, the utility values for the development events for the non-derelict version of the existing space type are calculated as specified previously and then the utility values for the development events for the derelict versions are calculated by applying the additional constants as specified here. That is, for example, the utility per unit land for the transition from existing derelict space of type v to new space of type h, $DRU_{v,h}$ is determined by first calculating the utility per unit land for the transition from existing space of type v, $RU_{v,h}$ as determined using Equations 51 through 53 shown above, and then applying the additional constants using Equation 63 shown above. The transitions from the derelict version of the space are then considered using $DRU_{v,h}$, $DRU_{v,E0}$, $DRU_{v,Er}$ and $DRU_{v,Er}$ for the 'new space type and quantity', 'no change', 'renovate' and 'demolish' events in the same way that the transitions from the non-derelict version of the space are considered using $RU_{v,h}$, $RU_{v,E0}$, $RU_{v,Er}$ and $RU_{v,Ex}$ as described below. Note that the dispersion parameters – the various λ parameters concerning the calculation of transition probabilities indicated below – are also different for the derelict and non-derelict cases for the existing space of a particular type, as discussed further below.

Table 7 provides a summary of the transition cost rates, utility function constants and treatments of age and quantity of updated space for each development event.

Table 7: Summary of transition cost rates, utility function constants and treatments of age and quantity of updated space for development events

Event Name		Transition cost from existing space type v to updated space type h per unit of new space type	Transition cost from existing space type v to updated space type h per unit of land	Transition constants	Updated Age	Updated quantity for space, $UArea_{h^*}$
new space type and quantity	En	$TrCostsS_{v,h}$	$TrCostsL_{v,h}$	$TrConst_{v,En}$ and $TrConst_{v,h}$ And also $DTrConst_{v,En}$ And $DTrConst_{v,h}$ when existing is derelict	0	Determined in second stage
no change	E0	0	0	$TrConst_{v,E0}$ And also $DTrConst_{v,E0}$ when existing is derelict	Existing age + 1	Existing quantity, $EArea_v$
add	Ea	$TrCostsS_{v,h=vEa}$	$TrCostsL_{v,h=vEa}$	$TrConst_{v,Ea}$	Average of existing age and 0	Additional amount determined in second stage
renovate	Er	$TrCostsS_{v,h=vEr}$	$TrCostsL_{v,h=vEr}$	$TrConst_{v,Er}$ And also $DTrConst_{v,Er}$ when existing	0	Existing quantity, $EArea_v$

				is derelict		
demolish	Ex	$TrCostsS_{v,h=vEx}$	$TrCostsL_{v,h=vEx}$	$TrConst_{v,Ex}$ And also $DTrConst_{v,Ex}$ when existing is derelict	0	0
derelict	Ed	0	0	$TrConst_{v,Ed}$	0	Existing quantity, $EArea_v$

The probability of each development event is then calculated for the parcel using the utility values per unit land indicated above and a nested logit structure as shown in Figure 6.

Groups of development events are defined for use in the calculation of these probabilities, as follows:

Gy = union of the 'new space type and quantity' (En) and 'add' (Ea) development events, which is the event that either development event En or development event Er occurs; this covers all instances where new space is developed

Gw = union of the 'demolish' (Ex) and 'derelict' (Ed) development events, which is the event that either development event Ex or development event Ed occurs; this covers all instances where the result is no space available for use after the transition

Gz = union of the 'new space type and quantity' (En), 'add' (Ea) and 'renovate' development events, which is the event that either development event En or development event Ea or development Er occurs; this covers all instances where the result is at least some new space available for use after the transition

Gk = union of all the development events other than 'no change' E0, which is the event that there is some change

These are indicated in Figure 6 at the corresponding higher-level nodes on the logit tree structure, consistent with assumptions about the comparatively similarities among the development events and the associated error terms in their utility functions.

The probabilities of the development events are calculated as follows:

$$Pr(h \mid En) = \exp(\lambda_{s,n} \cdot RU_{v,h}) / \Sigma_{h\varepsilon H(n)} \exp(\lambda_{s,n} \cdot RU_{v,h}) \quad \text{for each } h\varepsilon H(n) \tag{66}$$

$$CU_{En} = (1/\lambda_{s,n}) \cdot \ln [\Sigma_{h\varepsilon H(n)} \exp(\lambda_{s,n} \cdot RU_{v,h})] \tag{67}$$

$$Pr(En \mid Gy) = \exp(\lambda_{s,y} \cdot CU_{En}) / (\exp(\lambda_{s,y} \cdot CU_{En}) + \exp(\lambda_{s,y} \cdot RU_{v,Ea})) \tag{68}$$

$$Pr(Ea \mid Gy) = \exp(\lambda_{s,y} \cdot RU_{v,Ea}) / (\exp(\lambda_{s,y} \cdot CU_{En}) + \exp(\lambda_{s,y} \cdot RU_{v,Ea})) \tag{69}$$

$$CU_{Gy} = (1/\lambda_{s,y}) \cdot \ln [\exp(\lambda_{s,y} \cdot CU_{En}) + \exp(\lambda_{s,y} \cdot RU_{v,Ea})] \tag{70}$$

$$Pr(Er \mid Gz) = \exp(\lambda_{s,z} \cdot RU_{v,Er}) / (\exp(\lambda_{s,z} \cdot CU_{Gy}) + \exp(\lambda_{s,z} \cdot RU_{v,Er})) \quad (71)$$

$$Pr(Gy \mid Gz) = \exp(\lambda_{s,z} \cdot CU_{Gy}) / (\exp(\lambda_{s,z} \cdot CU_{Gy}) + \exp(\lambda_{s,z} \cdot RU_{v,Er})) \quad (72)$$

$$CU_{Gz} = (1/\lambda_{s,z}) \cdot \ln[\exp(\lambda_{s,z} \cdot CU_{Gy}) + \exp(\lambda_{s,z} \cdot RU_{v,Er})] \quad (73)$$

$$Pr(Ed \mid Gw) = \exp(\lambda_{s,w} \cdot RU_{v,Ed}) / (\exp(\lambda_{s,w} \cdot RU_{v,Ed}) + \exp(\lambda_{s,w} \cdot RU_{v,Ex})) \quad (74)$$

$$Pr(Ex \mid Gw) = \exp(\lambda_{s,w} \cdot RU_{v,Ex}) / (\exp(\lambda_{s,w} \cdot RU_{v,Ed}) + \exp(\lambda_{s,w} \cdot RU_{v,Ex})) \quad (75)$$

$$CU_{Gw} = (1/\lambda_{s,w}) \cdot \ln[\exp(\lambda_{s,w} \cdot RU_{v,Ed}) + \exp(\lambda_{s,w} \cdot RU_{v,Ex})] \quad (76)$$

$$Pr(Gz \mid Gk) = \exp(\lambda_{s,k} \cdot CU_{Gz}) / (\exp(\lambda_{s,k} \cdot CU_{Gz}) + \exp(\lambda_{s,k} \cdot CU_{Gw})) \quad (77)$$

$$Pr(Gw \mid Gk) = \exp(\lambda_{s,k} \cdot CU_{Gw}) / (\exp(\lambda_{s,k} \cdot CU_{Gz}) + \exp(\lambda_{s,k} \cdot CU_{Gw})) \quad (78)$$

$$CU_{Gk} = (1/\lambda_{s,k}) \cdot \ln[\exp(\lambda_{s,k} \cdot CU_{Gz}) + \exp(\lambda_{s,k} \cdot CU_{Gw})] \quad (79)$$

$$Pr(Gk) = \exp(\lambda_{s,t} \cdot CU_{Gk}) / (\exp(\lambda_{s,t} \cdot RU_{v,E0}) + \exp(\lambda_{s,t} \cdot CU_{Gk})) \quad (80)$$

$$Pr(E0) = \exp(\lambda_{s,t} \cdot RU_{v,E0}) / (\exp(\lambda_{s,t} \cdot RU_{v,E0}) + \exp(\lambda_{s,t} \cdot CU_{Gk})) \quad (81)$$

$$CU_{t} = (1/\lambda_{s,t}) \cdot \ln[\exp(\lambda_{s,t} \cdot RU_{v,E0}) + \exp(\lambda_{s,t} \cdot CU_{Gk})] \quad (82)$$

$$Pr(Ex) = Pr(Ex \mid Gw) \cdot Pr(Gw \mid Gk) \cdot Pr(Gk) \quad (83)$$

$$Pr(Ed) = Pr(Ed \mid Gw) \cdot Pr(Gw \mid Gk) \cdot Pr(Gk) \quad (84)$$

$$Pr(Er) = Pr(Er \mid Gz) \cdot Pr(Gz \mid Gk) \cdot Pr(Gk) \quad (85)$$

$$Pr(Ea) = Pr(Ea \mid Gy) \cdot Pr(Gy \mid Gz) \cdot Pr(Gz \mid Gk) \cdot Pr(Gk) \quad (86)$$

$$Pr(En) = Pr(En \mid Gy) \cdot Pr(Gy \mid Gz) \cdot Pr(Gz \mid Gk) \cdot Pr(Gk) \quad (87)$$

$$Pr(h) = Pr(En) \cdot Pr(h \mid En) \quad \text{for each } h \varepsilon H(n) \quad (88)$$

where:

$Pr(h)$ = probability of ‘new space type and quantity’ development event, En, with updated space type h selected

$H(n)$ = set of updated space types available in ‘new space type and quantity’ development event, En, as permitted by zoning rules, each indexed h

in software: this is specified at the parcel level, built-up from zoning condition information

$\lambda_{s,n}$ = utility function dispersion parameter for development choice among new space types alternatives hεH(n) conditional on event En

CU_{En} = composite utility for set of updated space type alternatives hεH(n) available with 'new space type and quantity' development event, En

Pr(E0) = probability of 'no change' development event, E0

Pr(Ex) = probability of 'demolish' development event, Ex

Pr(Ed) = probability of 'derelict' development event, Ed

Pr(Er) = probability of 'renovate' development event, Er

Pr(Ea) = probability of 'add' development event, Ea

Pr(Gw) = probability of event Gw, which is the union of the 'demolish' and 'derelict' development events, Ex and Ed

Pr(Gy) = probability of event Gy, which is the union of the the 'new space type and quantity' and 'add' development events, En and Ea

Pr(Gz) = probability of event Gz, which is the union of the 'new space type and quantity', 'add' and 'renovate' development events, En, Ea and Er

Pr(Gk) = probability of event Gk, which is the union of all the development events other than 'no change', En, Ea, Ex, Ed and Er

$Pr(En \mid Gy)$ = probability of development event En conditional on event Gy

$Pr(Ea \mid Gy)$ = probability of development event Ea conditional on event Gy

$\lambda_{s,y}$ = utility function dispersion parameter for development event choice between En and Ea conditional on event Gy

insoftware: sd.properties(DevelopmentAlternativesDispersionParameter);

CU_{Gy} = composite utility for event Gy, which is the union of development events En and Ea

$Pr(Er \mid Gz)$ = probability of development event Er conditional on event Gz

$Pr(Gy \mid Gz)$ = probability of event Gy conditional on event Gz

$\lambda_{s,z}$ = utility function dispersion parameter for development event choice between Er and Gy conditional on event Gz

insoftware: sd.properties(DevelopmentAlternativesDispersionParameter);

CU_{Gz} = composite utility for event Gz, which is the union of development events En, Ea and Er

$Pr(Ed \mid Gw)$ = probability of development event Ed conditional on event Gw

$Pr(Ex \mid Gw)$ = probability of development event Ex conditional on event Gw

$\lambda_{s,w}$ = utility function dispersion parameter for development event choice between Ed and Ex conditional on event Gw

insoftware: sd.properties(DevelopmentAlternativesDispersionParameter);

CU_{Gw} = composite utility for event Gw, which is the union of development events Ex and Ed

$Pr(Gz \mid Gk)$ = probability of event Gz conditional on event Gk

$Pr(Gw \mid Gk)$ = probability of event Gw conditional on event Gk

$\lambda_{s,k}$ = utility function dispersion parameter for development event choice between Gz and Gw conditional on event Gk
insoftware: sd.properties(DevelopmentAlternativesDispersionParameter);

CU_{Gk} = composite utility for event Gk, which is the union of all the development events other than 'no change'

CU_t = composite utility for union of all developer events

$\lambda_{s,t}$ = utility function dispersion parameter for development event choice between Gk and E0
insoftware: sd.properties(DevelopmentAlternativesDispersionParameter);

Note that the development events defined in the nested logit equations listed above with E_ designations and labels in boxes are mutually exclusive and collectively exhaustive, spanning the full possible event space. This is the same set of development events presented in Table 2 above: En, Ea, Er, Ed, Ex and E0.

The probabilities determined above are used in a Monte Carlo process to select the development event that occurs. If E0, Er or Ea is selected, then the updated space type h* is the same as the existing space type v. If Ed is selected, then the updated space type h* is the 'derelict' version of the existing space type v. If Ex is selected, then the parcel is deemed 'empty' and there is no space type. If En is selected, then the probabilities $Pr(h \mid En)$ for the set of $h \varepsilon H(n)$ permitted by the zoning rules are used in a further Monte Carlo process to select the updated space type h*.

The calculations for transitions where the existing space is derelict are identical to those set out immediately above, except that only the 'new space type and quantity', 'no change', 'renovate' and 'demolish' events are considered; the $DRU_{v,h}$, $DRU_{v,E0}$, $DRU_{v,Er}$ and $DRU_{v,Ex}$ utility values are used instead of the corresponding $RU_{v,h}$, $RU_{v,E0}$, $RU_{v,Er}$ and $RU_{v,Ex}$ utility values; and different values are used for the relevant dispersion parameters as follows:

$D\lambda_{s,n}$ = utility function dispersion parameter for development choice among new space types alternatives $h \varepsilon H(n)$ conditional on event En for the case where the existing space is derelict, replacing $\lambda_{s,n}$
insoftware: sd.properties(DevelopmentAlternativesDispersionParameter);

$D\lambda_{s,z}$ = utility function dispersion parameter for development event choice between Er and Gy conditional on event Gz for the case where the existing space is derelict, replacing $\lambda_{s,z}$
insoftware: sd.properties(DevelopmentAlternativesDispersionParameter);

$D\lambda_{s,k}$ = utility function dispersion parameter for development event choice between Gz and Gw conditional on event Gk for the case where the existing space is derelict, replacing $\lambda_{s,k}$
insoftware: sd.properties(DevelopmentAlternativesDispersionParameter);

$D\lambda_{s,t}$ = utility function dispersion parameter for development event choice between Gk and E0 for the case where the existing space is derelict, replacing $\lambda_{s,t}$
insoftware: sd.properties(DevelopmentAlternativesDispersionParameter);

Note that alternative parameters are not required for $\lambda_{s,y}$ or $\lambda_{s,w}$ for the case where the existing space is derelict – because the corresponding elements of the logit model nesting structure are eliminated with the elimination of the 'add' and 'demolish' development events when the existing space is derelict. So there is no $D\lambda_{s,y}$ or $D\lambda_{s,w}$ parameter.

Empty *Parcel as a Special Case*

The 'empty' designation for a parcel indicates there is no developed space on the parcel. Such a parcel might also be termed 'undeveloped', 'greenfield', 'raw' or even 'brownfield', and it may or may not have water, power, telephone or other services available. The essential point is that there is no developed space on the parcel. This 'empty' state is a standard, universal condition that must be included as a possible state for parcels in every model – because there are always some portions of land that are or could be empty in any area being modelled. All the developed space types are defined as part of the specific model design for a particular application, and in general vary from one model application to the next. The 'empty' designation is different; it is the absence of developed space, and is not a space type like any other – and thus is a special designation that is always included in every model.

Development events involving existing empty parcels are handled in the SD Module in much the same way that development events involving existing space are handled as described above. There are some special aspects. Demolition costs are set to 0,

consistent with there being no existing space. Any clean-up costs, as may arise with 'brownfield' sites, are included in the construction costs for the updated space. The 'add', 'renovate', 'demolish' and 'derelict' development events are not possible. But the calculation of the utility values and the transition probabilities for the 'new space type and quantity' and the 'no change' events are otherwise the same as described above.

In spite of these similarities in the treatment of transitions, the special nature of the existing empty designation for a parcel leads to the separate specification of the relevant transition constants and other utility function parameters for empty parcels in the input files. All of the defined space types are considered in one set of input files, whereas the existing empty category is considered separately in another input file.

Treatment of Lands Used Directly by Activities

Some activities use land directly, rather than building space on land. For example, agricultural and extraction activities both draw directly from the land on which they are located. In the PECAS framework these activities use some form of developed space – such as 'Agricultural Space' or 'Mining Space' – that reflects some form of development action where the corresponding land has been prepared for use – where fences have been built or equipment has been installed. This preserves the theoretical structure of the PECAS framework where activities in the AA Module use developed space and developer actions in the SD Module transform the space on parcels of land and thereby provide the quantities of developed space for activities to use. Thus, the treatment of parcels of land supporting agricultural activities is analogous to the treatment of parcels of land containing residential or commercial activities.

Second Stage: Selecting Updated Space Quantity

In the second stage the updated space quantity is selected conditional on the development event and updated space type h* selected in the first stage.

'*No Change', 'Renovate' or 'Derelict' Development Events*

If one of the 'no change', 'renovate' or 'derelict' development events, E0, Er and Ed, respectively, is selected in the first stage, then the updated space quantity, $UArea_{h^*}$, is the same as the existing space quantity, $EArea_v$.

'*New Space Type and Quantity' Development Event*

If the 'new space type and quantity' development, En, is selected in the first stage, then the updated space quantity, $UArea_{h^*}$, is determined using a Monte Carlo selection process.

A set of two continuous logit model functions are used in combination to establish the probability density function, pdf, for the distribution of selection probabilities over the range of permitted space quantities from $MinArea_{h^*}$ to $MaxArea_{h^*}$. This pdf is used as the sampling distribution in the selection of a specific space quantity, $UArea_{h^*}$, using the Monte Carlo process.

Two component logit model functions are used in combination to form this pdf. This is done in order to facilitate calibration of the resulting aggregate quantities of space development by type. The general shape of resulting pdf is shown in Figure 7. The calibration process involves adjusting the values for $StepPt_{h^*}$, $StepValue_{h^*}$, $SlopeLeft_{h^*}$ and $SlopeRight_{h^*}$.

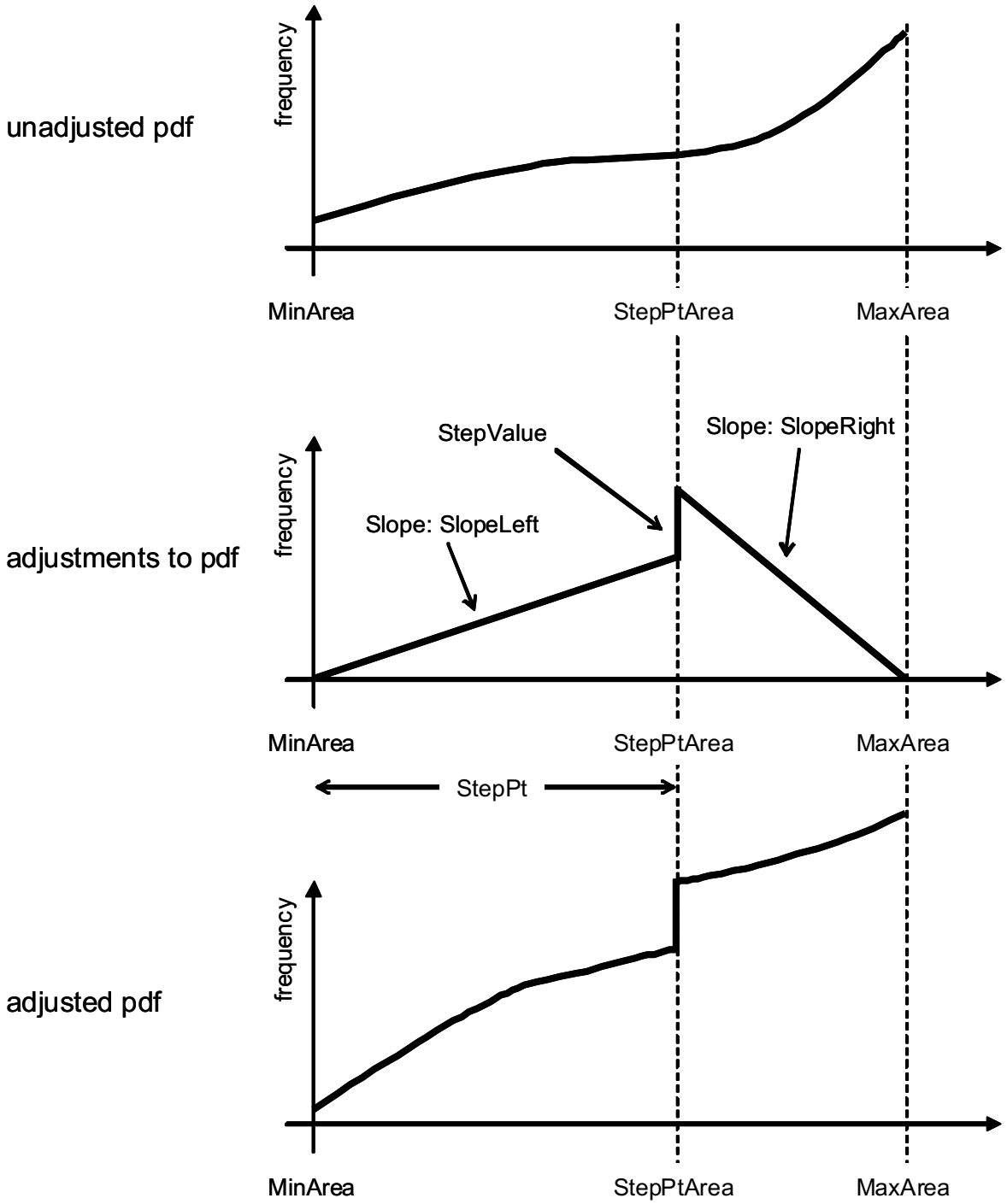

Figure 7: Form of adjustments to pdf for the distribution of selection probabilities over the range of permitted space quantities, using values for $StepPt_{h^*}$, $StepValue_{h^*}$, $SlopeLeft_{h^*}$ and $SlopeRight_{h^*}$.

The lower component logit model function for the pdf, indicating the distribution of selection probabilities for $UArea_{h^*}$, over the range of values for $UArea_{h^*}$ from $MinArea_{h^*}$ to $StepPtArea_{h^*}$, has the following form:

$$pdfL(UArea_{h^*}) = NumL_{h^*} / \{ IntL_{h^*} \cdot LotSize/(\lambda_{q,h^*} \cdot NetRev_{v,h^*}) \} \quad (89)$$

with:

$$NumL_{h^*} = \exp[\lambda_{q,h^*} \cdot ((NetRev_{v,h^*} \cdot UArea_{h^*})/LotSize - TrCostsL_{v,h^*}) + (SlopeLeft_{h^*} \cdot (UArea_{h^*} - MinArea_{h^*}) / LotSize)] \quad (90)$$

and

$$IntL_{h^*} = \exp[\lambda_{q,h^*} \cdot ((NetRev_{v,h^*} \cdot StepPtArea_{h^*})/LotSize - TrCostsL_{v,h^*}) + (SlopeLeft_{h^*} \cdot (StepPtArea_{h^*} - MinArea_{h^*}) / LotSize)] - \exp[\lambda_{q,h^*} \cdot ((NetRev_{v,h^*} \cdot MinArea_{h^*})/LotSize - TrCostsL_{v,h^*})] \quad (91)$$

and

$$StepPtArea_{h^*} = MinArea_{h^*} + StepPt_{h^*} \cdot (MaxArea_{h^*} - MinArea_{h^*}) \quad (92)$$

where:

$UArea_{h^*}$ = quantity of updated space type h* for parcel
in software: this is recorded at the parcel level into the parcel database

$NumL_{h^*}$ = function numerator term for lower component of pdf($UArea_{h^*}$) for space type h*; Num_{h^*} is itself a function of $UArea_{h^*}$ and thus could be written $Num_{h^*}(UArea_{h^*})$

$IntL_{h^*}$ = function denominator term for lower component of pdf($UArea_{h^*}$) for space type h*; $IntL_{h^*}$ is itself a function of $UArea_{h^*}$ and thus could be written $IntL_{h^*}(UArea_{h^*})$

$MinArea_{h^*}$ = the lower bound on the range of permitted space quantities considered for $UArea_{h^*}$ for the parcel;

$StepPtArea_{h^*}$ = the boundary value between the lower component and the upper component of the range of permitted space quantities considered for $UArea_{h^*}$ for the parcel;

$SlopeLeft_{h^*}$ = function parameter included in pdf($UArea_{h^*}$) for space type h*; used in calibration of the aggregate quantity of space development of space type

h*, this parameter is the slope of the calibration portion of the function over the component of pdf($UArea_{h^*}$) to the left of (or below) $StepPtArea_{h^*}$

$StepPt_{h^*}$ = function parameter included in pdf($UArea_{h^*}$) for space type h*; used to specify the location of the $StepPtArea_{h^*}$ boundary value, expressing it as a proportion out from $MinArea_{h^*}$ from along the range from $MinArea_{h^*}$ to $MaxArea_{h^*}$.

The upper component logit model function for the pdf, over the range of values for $UArea_{h^*}$ from $MidPtArea_{h^*}$ to $MaxArea_{h^*}$, has the following form:

$$pdfU(UArea_{h^*}) = Num_{h^*} / \{ IntU_{h^*} \cdot LotSize/(\lambda_{q,h^*} \cdot NetRev_{v,h^*}) \} \quad (93)$$

with:

$$Num_{h^*} = \exp[\lambda_{q,h^*} \cdot ((NetRev_{v,h^*} \cdot UArea_{h^*})/LotSize - TrCostsL_{v,h^*}) + (SlopeLeft_{h^*} \cdot (StepPtArea_{h^*} - MinArea_{h^*}) / LotSize) + StepValue_{h^*} + (SlopeRight_{h^*} \cdot (UArea_{h^*} - StepPtArea_{h^*}) / LotSize)] \quad (94)$$

and

$$IntU_{h^*} = \exp[\lambda_{q,h^*} \cdot ((NetRev_{v,h^*} \cdot MaxArea_{h^*})/LotSize - TrCostsL_{v,h^*}) + (SlopeLeft_{h^*} \cdot (StepPtArea_{h^*} - MinArea_{h^*}) / LotSize) + StepValue_{h^*} + (SlopeRight_{h^*} \cdot (MaxArea_{h^*} - StepPtArea_{h^*}) / LotSize)] - \exp[\lambda_{q,h^*} \cdot ((NetRev_{v,h^*} \cdot StepPtArea_{h^*})/LotSize - TrCostsL_{v,h^*}) + (SlopeLeft_{h^*} \cdot (StepPtArea_{h^*} - MinArea_{h^*}) / LotSize) + StepValue_{h^*}] \quad (95)$$

where:

$NumU_{h^*}$ = function numerator term for upper component of pdf($UArea_{h^*}$) for space type h*; $NumU_{h^*}$ is itself a function of $UArea_{h^*}$ and thus could be written $NumU_{h^*}(UArea_{h^*})$

in software: this is recorded at the parcel level into the parcel database

$IntU_{h^*}$ = function denominator term for upper component of pdf($UArea_{h^*}$) for space type h*; $IntU_{h^*}$ is itself a function of $UArea_{h^*}$ and thus could be written $IntU_{h^*}(UArea_{h^*})$

$MaxArea_{h^*}$ = the upper bound on the range of permitted space quantities considered for $UArea_{h^*}$ for the parcel;

$StepValue_{h^*}$ = function parameter included in pdf($UArea_{h^*}$) for space type h*; used in calibration of the aggregate quantity of space development of space type h*, this parameter is the size of the discontinuity in the calibration portion of the function at the point where $UArea_{h^*}$ is $StepPtArea_{h^*}$, the boundary between the left (below) and right (above) components of pdf($UArea_{h^*}$)

$SlopeRight_{h^*}$ = function parameter included in pdf($UArea_{h^*}$) for space type h*; used in calibration of the aggregate quantity of space development of space type h*, this parameter is the slope of the calibration portion of the function over the component of pdf($UArea_{h^*}$) to the right of (or above) $StepPtArea_{h^*}$

Equations 87 through 93 define the pdf used to select a value for $UArea_{h^*}$. More specifically, the mechanical steps in the Monte Carlo process are to first select a value r for a random variable R that has a uniform sampling distribution over the interval [0,1] and then determine the value for $UArea_{h^*}$ that provides the same value as r from the cumulative distribution form of this pdf.

Rents

The expected rent for each updated space type h on the parcel, $Rent_h$, is calculated taking into account the zonal-level price established for the current year in the AA Module and local-level effects due to the density of development around the parcel, the distance from (or proximity to) local-level influences, and the age of the existing space on the parcel, if any.

The expected rent for updated space type h is calculated as follows:

$$Rent_h = Price_{h,z} \cdot \pi_{g \varepsilon G} LEFac_{g,h} \quad (96)$$

where:

$Rent_h$ = rent for updated space type h on parcel; in units of money per unit of area for space type h per year;

$Price_{h,z}$ = price for updated space type h determined in AA Module for current year; in units of money per unit of area for space type h per year;
in software: ExchangeResults(Price);

g = index of local-level effects on rent
in new PA-database version of software: LocalLevelEffects(LLEffectID);

G = set of all local-level effects on rent considered

π = the series multiplication operator, it indicates a series of terms are multiplied together in the same way that the series addition operator, Σ, indicates a series of terms are added together

$LEFac_{g,h}$ = factor adjusting proportional change in rent for space type h as a function of values on dimension relevant for local-level effect g
in new PA-database version of software: LocalLevelEffects(LLEffectFactor);

There are several different forms available for the function calculating values for $LEFac_{g,h}$ as follows:

1: Constant:

$$LEFac_{g,h} = \theta_g \text{ if } DValue_g = RefDValue_g \text{ is true and 1 otherwise} \quad (97)$$

2: Exponential: ***WAS 2 IN PREVIOUS DOCUMENTATION***

$$LEFac_{g,h} = \exp\left(\theta_g \cdot \{ DValue_g / RefDValue_g \} \right) \quad (98)$$

3: Shifted Exponential: ***WAS 3 IN PREVIOUS DOCUMENTATION***

$$LEFac_{g,h} = \exp\left(\theta_g \cdot [1 - \{ DValue_g / RefDValue_g \}] \right) \quad (99)$$

4: Power: ***WAS 1 IN PREVIOUS DOCUMENTATION***

$$LEFac_{g,h} = \left(\theta_g \right)^{(DValue_g / RefDValue_g)} \quad (100)$$

5: Shifted Power:

$$LEFac_{g,h} = \left(\theta_g \right)^{(1 - \{ DValue_g / RefDValue_g \})} \quad (101)$$

6: Reversed Power:

$$LEFac_{g,h} = \left(1 - \theta_g \right)^{(DValue_g / RefDValue_g)} \quad (102)$$

7: Reversed Shifted Power:

$$LEFac_{g,h} = (1 - \theta_g)^{(1 - \{ DValue_g / RefDValue_g \})} \quad (103)$$

8: Multiplicative: ***WAS 4 IN PREVIOUS DOCUMENTATION***

$$LEFac_{g,h} = (DValue_g / RefDValue_g)^{\theta g} \quad (104)$$

9: Shifted Multiplicative:

$$LEFac_{g,h} = (1 - \{ DValue_g / RefDValue_g \})^{\theta g} \quad (105)$$

where:

$DValue_g$ = values on dimension relevant for local-level effect g – typically this is representing the distance from the parcel to the source of the local-level effect, the local-level density for the parcel, or the age of the space on the parcel
in software: this is determined at the parcel level, built-up from parcel-level database information using a PA-Macro

$RefDValue_g$ = reference value on dimension relevant for local-level effect g
in software: LocalLevelEffectsI(MaximumDistanceEffect);

θ_g = parameter for function calculating values for $LEFac_{g,h}$
in software: currently ;

For each local-level effect, the value of $DValue_g$ where the value of $LEFac_{g,h}$ equals 1 defines the 'reference point' where the zonal-level space price, $Price_{h,z}$, applies and the particular local-level effect has no impact. $RefDValue_g$ can be used to set a maximum, minimum or reference value for $DValue_g$ in estimation as well as in application.

These local-level adjustment factors provide a flexible set of terms for representing the different impacts on rents arising from local-level effects, including the age of the current space on the parcel, the 'local-area' density in the immediate vicinity of the parcel and the proximity of the parcel to both attractive and unattractive items such as local green spaces, beaches, freeways, freeway ramps, major roads, schools, and transit hubs.

The measures of age, local-area density and distance used for $DValue_g$ for each of the gεG are calculated for each parcel exogenously (in the parcel database and the GIS and database operations related to it) and then passed to the SD Module as inputs.

Some of the specific functional forms are discussed briefly immediately below. Detailed descriptions of the full set of functional forms in Equations 97 through 105 inclusive together with instructions on how to develop and use them are included the 'Rent Modifier Equations' System Documentation Technical Note included with the Model Development Manual.

Function 1: Constant can be used to apply specific constants for particular parcels in certain cases. That is, it can be used to apply a constant value $c = \theta_g$ to specific parcels in certain cases (when $DValue_g = RefDValue_g$) and a value 1 to all other parcels.

Function 2: Exponential is typically used to represent the impacts of local-level effects on rents where the distance from the source of the effect alters the strength of its impact. In these cases, $DValue_g$ is the distance from the source of the effect to the parcel. The 'reference point' where $LEFac_{g,h}$ is 1 arises where $DValue_g$ is 0, that is, when the source of the effect is located right at the parcel. Negative values for θ_g result in values for $LEFac_{g,h}$ that decrease from 1 (but do not go below 0) as $DValue_g$ increases from 0. Thus, rents decrease down from the zonal-level value with increasing distance away from the effect. Examples of local-level effects where this would be appropriate are:

- local access effects: The distance to the nearest freeway entrance can have important effects on the ease of access to and thus on the attractiveness of the parcel for manufacturing and distribution activities, which can affect the rent for industrial, warehouse and commercial space; with the attractiveness and thus the rent decreasing from the reference value as distance increases; and
- frontage effects: The distance to the nearest roadway, including whether the parcel fronts right on the roadway, can have important effects on the availability of pass-

by traffic and thus on the attractiveness of the parcel for retail activities, which can affect the rent for retail and commercial space; with the attractiveness and thus the rent decreasing from the reference value as distance increases.

Function 3: Shifted Exponential is also typically used to represent the impacts of local-level effects on rents where the distance from the source of the effect alters the strength of its impact. Again, in these cases, $DValue_g$ is the distance from the source of the effect to the parcel, but the 'reference point' where $LEFac_{g,h}$ is 1 arises where $DValue_g$ is equal to $RefDValue_g$, that is, when the source of the effect is a distance $RefDValue_g$ from the parcel. Positive values for θ_g result in values for $LEFac_{g,h}$ that increase from 1 as $DValue_g$ decreases from $RefDValue_g$ to 0; thus, rents increase up from the zonal-level value as the effect gets closer to the parcel. Negative values for θ_g result in values for $LEFac_{g,h}$ that decrease from 1 as $DValue_g$ decreases from $RefDValue_g$ to 0; thus, rents decrease down from the zonal-level value as the effect gets closer to the parcel. Examples of local-level effects where this decrease from a reference value would be appropriate are:

- nuisance effects: the distance to nearest freeway or major roadway can have important effects on noise and air quality and thus on the attractiveness of the parcel for household activities, which can affect the rent for residential space types; with the attractiveness and thus the rent decreasing from the reference value as the distance decreases.

Function 4: Power is typically used to represent the impacts of local-area density and the age of the existing space on rents. Values for θ_g between 0 and 1, and typically close to 0, result in values for $LEFac_{g,h}$ that lie between 0 and 1 and decrease as $DValue_g$ increases. In such cases, using local-area density for $DValue_g$ results in reductions in rents as density increases, consistent with expectations; similarly, using age for $DValue_g$ results in reductions in rents as age increases, again consistent with expectations. In both cases, the 'reference point' where $LEFac_{g,h}$ is 1 arises when $DValue_g$ (as the local-

area density or the age) is 0. Note that the age of the space on the parcel, Age_h, is an influence on rent (taking a value other than 0) only for the 'No Change' development event.

Amortization Factor

The choice models in the SD Module consider conditions for a one-year period, with utilities, prices, rents and all other money values expressed in terms of the amount per year. Principal values such as construction and demolition costs are amortized into the corresponding yearly amounts before they are input to the choice models. This is done by multiplying the principal amount by an amortization factor. The resulting product is the equivalent yearly value. The amortization factor is calculated as follows:

$$Amt(Yrs,Int) = Int / [1 – (1/(1+Int)^{Yrs})] \quad (106)$$

where:

Amt(Yrs,Int) = amortization factor adjusting principal unit cost into equivalent yearly payments for number of years Yrs and interest rate Int
in software: sd.properties(AmortizationFactor)

Yrs = number of years for amortizing principal unit costs, typically the standard 'lifetime' of development projects
in new PA-database version of software: sd.properties(Years), replacing direct specification of AmortizationFactor as indicated above

Int = interest rate as a proportion, that is, 8% is expressed as 0.08;
in new PA-database version of software: sd.properties(InterestRate) , replacing direct specification of AmortizationFactor as indicated above

Transition Costs Per Unit Updated Space

The amortized money cost for transitioning from existing space type v to updated space type h on the parcel, $TrCostsS_{v,h}$, includes all cost elements for this transition that vary linearly with the quantity of updated space. It is calculated as follows:

$$TrCostsS_{v,h} = Amt(Yrs,Int) \cdot (InitCostsS_h) + OnCostsS_h \quad (107)$$

where:

$InitCostsS_h$ = principal unit costs for transitioning to space type h on parcel, expressed as an amount per unit of space type h;
in software: DevelopmentTypesI(ConstructionCost) plus ZoningSchemes(SpaceFee);
in new PA-database version of software: this is specified at the parcel level, built-up from parcel level information on fees, construction costs and zoning conditions penalty;

$OnCostsS_h$ = ongoing unit costs for transitioning to space type h on parcel, expressed as an amount per unit of space type h per year;
in new PA-database version of software: this is specified at the parcel level, built-up from parcel level information on fees, construction costs and zoning conditions penalty;

The components of cost included in $InitCostsS_h$ values are typically:

- unit construction costs for space type h on parcel, expressed as the rate per unit of space type h; established for each parcel using model-wide unit construction costs possibly modified to reflect parcel-specific cost influences such as groundwater or slope;
- costs for improving services as required for space type h on parcel, expressed as the rate per unit of space type h; established for each parcel taking into account existing services and the improvements required for each additional unit of space type h; services here can include water, sewer and power; note that elements of these costs should not be included both here and in $InitCostsL_{v,h}$;
- any one-time, up-front fees (such as certain development fees) that are charged per unit of space for development of space type h on parcel, expressed as the rate per unit of space type h; and
- any subsidies that are paid one-time, up-front per unit of space for development of space type h on parcel, expressed as a negative value of cost and as the rate per unit of space type h.

The components of cost included in $OnCostsS_h$ values are typically:

- any fees or taxes that are charged on an ongoing basis per unit of space for space type h on parcel, expressed as the rate per unit of space type h per year; and

- any subsidies that are paid on an ongoing basis per unit of space for space type h on parcel, expressed as a negative value of cost and as the rate per unit of space type h per year.

The components of the $InitCostsS_h$ and $OnCostsS_h$ values are calculated and summed for each parcel using a GIS with special command macros established as part of the development of a specific PECAS model. Note that elements of these costs should not be included in these values and also in the $InitCostsL_{v,h}$ and $OnCostsL_h$ values described below; any elements that are included in both the 'per unit of space' and the 'per unit of land' values are double-counted.

Transition Costs Per Unit Land Area

The amortized money cost for transitioning from existing space type v to updated space type h on the parcel, $TrCostsL_{v,h}$, includes all cost elements for this transition that vary linearly with the quantity of land on the parcel. It is calculated as follows:

$$TrCostsL_{v,h} = Amt(Yrs,Int) \cdot (InitCostsL_{v,h}) + OnCostsL_h \quad (108)$$

where:

$InitCostsL_{v,h}$ = principal unit costs for transitioning from space type v to space type h on parcel, expressed as an amount per unit of land on parcel;
in software: DevelopmentTypesI(LandFee);
in new PA-database version of software: this is specified at the parcel level, built-up from parcel level information on fees, construction costs and zoning conditions penalty;

$OnCostsL_h$ = ongoing unit costs for transitioning to space type h on parcel, expressed as an amount per unit of land h per year;
in new PA-database version of software: this is specified at the parcel level, built-up from parcel level information on fees, construction costs and zoning conditions penalty;

The components of cost included in $InitCostsL_{v,h}$ values are typically:

- costs for demolition of existing space, if any, on parcel, expressed as the rate per unit of land; established for each parcel using model-wide unit demolition costs for

space type v multiplied by the quantity of existing space of type v and then divided by the land area of the parcel;

- costs for improving services as required for space type h on parcel, expressed as the rate per unit of land; established for each parcel taking into account existing services and the improvements required for each additional unit of space type h; services here can include water, sewer and power; note that elements of these costs should not be included both here and in $InitCostsS_h$;
- any one-time, up-front fees (such as certain development fees) that are charged per unit of land for development of space type h on parcel, expressed as the rate per unit of land; and
- any subsidies that are paid one-time, up-front per unit of land for development of space type h on parcel, expressed as a negative value of cost and as the rate per unit of land.

The components of cost included in $OnCostsL_h$ values are typically:

- any fees or taxes that are charged on an ongoing basis per unit of land for space type h on parcel, expressed as the rate per unit of land per year; and
- any subsidies that are paid on an ongoing basis per unit of land for space type h on parcel, expressed as a negative value of cost and as the rate per unit of land per year.

The components of the $InitCostsL_{v,h}$ and $OnCostsL_h$ values are calculated and summed for each parcel using a GIS with special command macros established as part of the development of a specific PECAS model. As indicated above: elements of these costs should not be included in these values and also in the $InitCostsS_h$ and $OnCostsS_h$ values described above; any elements that are included in both the ‘per unit of space’ and the ‘per unit of land’ values are double-counted.

Maintenance Costs

The maintenance costs for updated space type h on parcel, $MtCostsS_h$, are calculated taking into account the age of the space. For any new space the maintenance costs are zero. Thus, maintenance costs apply only for the 'no change' or 'add' development events, where all or part of the space is retained as is – with the same type, the same quantity, and has a non-zero age equal to the age of the existing space plus one more year. The maintenance costs per unit of updated space type h are calculated as follows:

$$MtCostsS_h = MtBase_h (1+MtFac_h)^{(Age-1)} \quad \text{for 'no change' development event} \quad (109)$$

$$= 0 \quad \text{otherwise}$$

where:

$MtBase_h$ = base unit costs for maintaining space type h (when it has an age of 1 year), expressed as a money amount per unit of space of type h for a year;
in software: DevelopmentTypesI(MaintenanceCost);
in new PA-database version of software: SpaceTypes(MaintenanceCost);

$MtFac_h$ = proportional increase in maintenance costs for space type h with each additional year of age;
in software: DevelopmentTypesI(AgeMaintenanceCost)
in new PA-database version of software: SpaceTypes(AgeMaintenanceCost);

Age = age of existing space on parcel in years;
in software: this is determined at the parcel level, built-up from parcel-level database information using a PA-Macro

Note that the age of the updated space on the parcel is reset to 0 for all development events other than 'no change'. Even the 'alter or renovate' development event, Er, results in the age of the updated space being reset to 0, regardless of the quantities of existing and updated space involved on the parcel.

6.4. Aggregation

When the consideration of the full list of all parcels is completed in a given year-to-year step, and all of the development events for those parcels have been assigned, then the updated quantities of space of each type in each TAZ and LUZ are calculated, as follows:

$$SpcQty_{h,z} = \Sigma_{p\varepsilon P(z)} UArea_{h,p} \quad \text{for all } z\varepsilon Z \quad \text{for each h} \qquad (110)$$

and

$$TSpcQty_{h,i} = \Sigma_{p\varepsilon P(i)} UArea_{h,p} \quad \text{for all } i\varepsilon I \quad \text{for each h} \qquad (111)$$

where:

z	=	index for land use zones (LUZ);
Z	=	set of all land use zones;
i	=	index for transport analysis zones (TAZ);
I	=	set of all transport analysis zones;
p	=	index for parcels;
P(z)	=	set of all parcels in land use zone z;
P(i)	=	set of all parcels in transport analysis zone i;
$UArea_{h,p}$	=	updated quantity of space type h in parcel p; in units specified for space type h;
$SpcQty_{h,z}$	=	updated aggregate quantity of space type h in land use zone z;
$TSpcQty_{h,i}$	=	updated aggregate quantity of space type h in transport analysis zone i;

$SpcQty_{h,z}$ and $TSpcQty_{h,i}$ are the quantities of space used in the AA Module and the TR Module for the next year (t+1).

6.5. Microsimulation Error

With the Monte Carlo process used in the SD Module, the aggregate quantities of space $SpcQty_{h,z}$ and $TSpcQty_{h,i}$ are random variables (actually 'estimators') with sampling distributions. In general, a different run provides different realized values for these random variables. This is not a practical concern when the variances on the sampling distributions are relatively low, but there is no guarantee that these variances will be relatively low in a given instance.

The standard deviation of the sampling distribution for one of these aggregate quantities of space is called the 'microsimulation error' for the quantity. The variances on the sampling distributions – effectively measured using the microsimulation error – need to be considered as part of the development and implementation of a PECAS model. This is done by running the model system multiple times and using the range of results to calculate the microsimulation error for each quantity of interest. These microsimulation errors can propagate over the series of years considered by the model and tend to magnify, so the quantities of interest for later years are particularly susceptible.

When the microsimulation error is deemed to be too high, there are a number of strategies that can be used to reduce it, as follows:

- consider more aggregate-level values;
- use larger samples, by working with smaller parcels or larger zones
- fix the random seed used in the Monte-Carlo calculations – this is in effect turning-off the random component and thereby not using a Monte-Carlo approach; this is particularly dangerous and fraught with errors in practice.
- add a fixed random term (sometimes called a 'prize' value) to the utility function for each development option for each parcel, drawing these values from a distribution

across the set of parcels, and these prize values are kept constant between comparable runs; or

- use the aggregate version of the SD Module, as described below.

6.6. Processing Algorithm

A flowchart of the processes used in the SD Module is shown in Figure 8, including the steps in the consideration of each parcel one-by-one and the information flows among the module components.

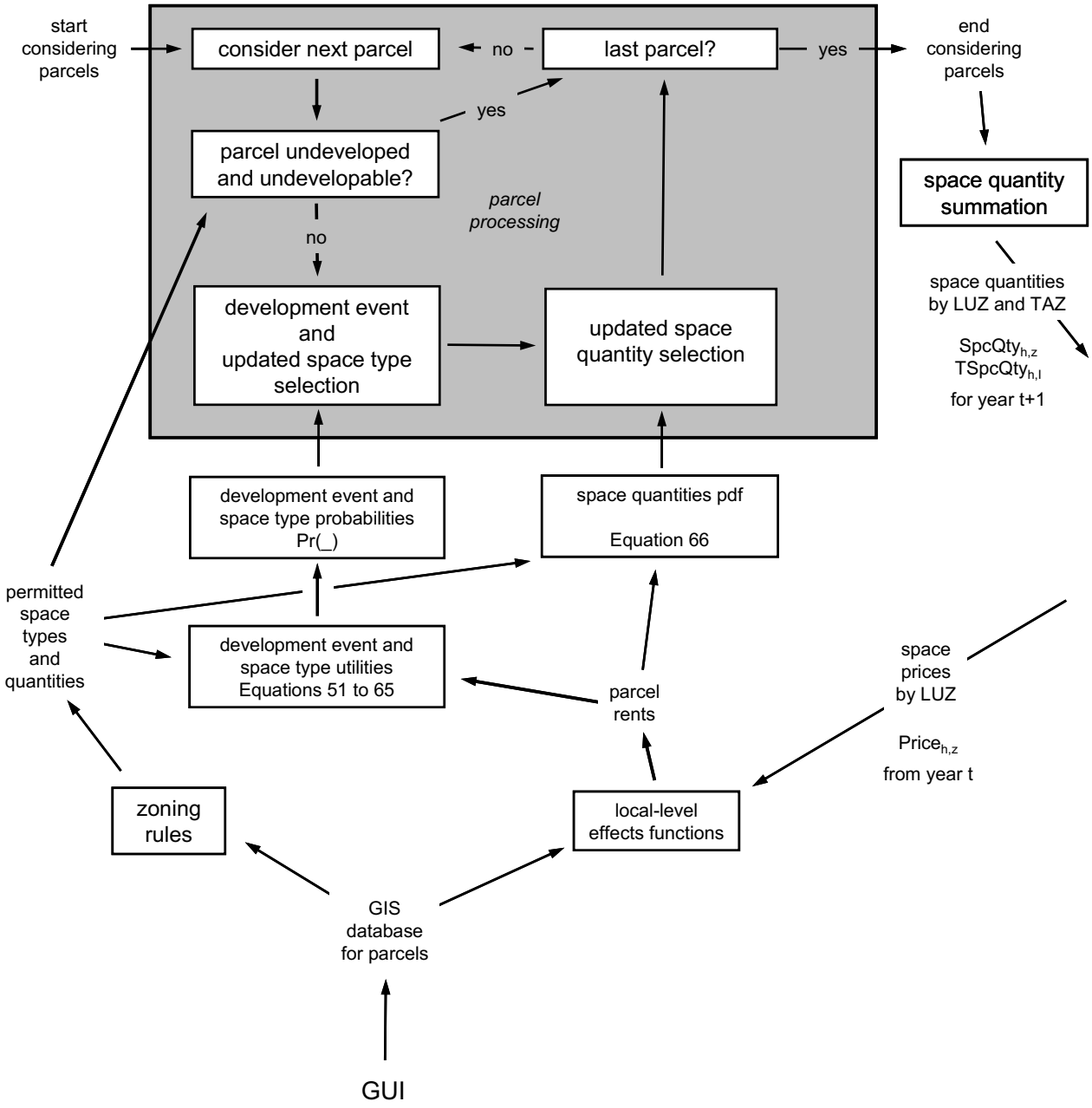

Figure 8: Steps and information flows in the SD Module, including the processing of each parcel one-by-one.

6.7. Pseudo-Parcels

Legal parcels of land – as opposed to model grid cells – in general vary in size and can sometimes be very large. In the SD Module, a virtual subdividision process called "pseudo-parcelling" can be applied to any parcel larger than a specified maximum land area, designated ParcelMax. The parcel is considered to be set of set of mutually exclusive and collectively exhaustive 'pseudo-parcels'. Each such pseudo-parcel is treated the same as a parcel in all the processing.

In the current version of PECAS, the "Parcel Cut" GIS functionality is used to pre-subdivide large parcels. Large parcels need to be split either using Parcel Cut or pseudo-parceling for several reasons:

- development can occur in phases over a period of years as a series of discrete development events on the pseudo-parcels of a parcel instead of all at once on the parcel;
- different development events can occur on the pseudo-parcels of a parcel giving rise to a mix of land uses for the parcel; and
- a larger sample size of events is considered in the microsimulation, which can help avoid difficulties with microsimulation error.

6.8. Constraining Development Activity through Construction Control

In some cases, depending on the model design, the AA Module output includes an indication of the amount or amounts of space constructed in the model area over the time period covered by a run of the SD Module. For example, the run of the AA Module for the year t could include an estimate of the quantity of residential space produced by the construction industry over that year, in general as the quantity of one or more puts made by one or more relevant building construction activities in the AA Module. This estimate can be used to inform (or control) the quantity of residential space added in the corresponding run of the SD Module for the period from year t to year t+1, and the results of the run of the SD Module in turn used to inform the representation of the corresponding construction activity for the next time period.

Deterministic Microsimulation Control

If the aggregate quantity of development activity in the SD Module is to be constrained, it is recommended to use the Deterministic Microsimulation version of SD. After the expected values of development are determined by summing the expected values across all the parcels in each LUZ, the total amount is scaled up or down across all the LUZs to match the construction constraint. Then, the resulting modified amount is assigned to the best parcels in each LUZ. This is described more in (Hunt et al., 2019).

Monte-Carlo Construction Control

In the Monte-Carlo implementation of SD, construction control works through a dynamic utility adjustment as parcels are considered in a random order. The processing algorithm as described in Section 6.6 considers the full list of parcels in random order, and the rates of development are compared with the constraints and adjusted as the processing continues in order to arrive at a result consistent with the constraints at the end of the processing. This simulates developers reacting to a short run construction cost function, where high rates of construction that strain capacities result in construction cost increases that slow rates of construction, and low rates of construction that leave capacity idle result

in construction cost decreases that increase rates of construction, working towards a final aggregate quantity of construction consistent with the cost function.

A set of target quantities of space construction are defined from the results of the run of the AA Module for year t as follows:

$$QConsTarget_k = \Sigma_{c\varepsilon C(k)} TP_c \cdot CFact_{c,k} \quad \text{for each } k\varepsilon K \qquad (112)$$

where:

- k = index for group of space types related to a specific target quantity of space construction;
- K = set of groups of space types related to target quantities of space construction;
- $C(k)$ = set of puts in the AA Module that are in the group of space types k related to a specific target quantity of space construction;
- $QConsTarget_k$ = target quantity of space construction for group of space types k; in units of physical space area (such as ft^2 or m^2)
- TP_c = quantity of put c produced by all activities in entire model area;
- $CFact_{c,k}$ = factor converting quantity in units for put c into quantity in units of space used for target quantity of construction for group of space types k

The treatment is the same for each of these target quantities as the full set of parcels is processed by the SD Module.

The parcels are processed in N batches, each batch containing another 1/N of the full set of parcels. These batches are indexed n where n goes from 1 to N. For each batch, the construction costs are multiplied by factors that adjust the construction cost values for each potential space transition for each parcel arising from the parcel database, as follows:

$$AdjTrCostsS_{v,h,n} = TrCostsS_{v,h} \cdot CostFact_{h,k,n}$$

and

$$AdjTrCostsL_{v,h,n} = TrCostsL_{v,h} \cdot CostFact_{h,k,n} \qquad (113)$$

where:

n = index for specific batch of 1/N of the full set of parcels; from n=1 for the first batch to n=N for the last batch;

$CostFact_{h,k,n}$ = cost adjustment factor for the construction costs for updated space type h related to the target for the group of space types k batch n of the full set of parcels

$AdjTrCostsS_{v,h,n}$ = the adjusted value for the amortized money cost for transitioning from existing space type v to updated space type h on the parcel, for all cost elements for this transition that vary linearly with the quantity of updated space, to use in place of $TrCostsS_{v,h}$ for the specific batch n of the full set of parcels;

$AdjTrCostsL_{v,h,n}$ = the adjusted value for the amortized money cost for transitioning from existing space type v to updated space type h on the parcel, for all cost elements for this transition that vary linearly with the quantity of land on the parcel, to use in place of $TrCostsL_{v,h}$ for the specific batch n of the full set of parcels;

After each batch of 1/N parcels is considered, the values for the $CostFact_{h,k,n}$ are updated in order to change the rates of space transitions consistent with the targets $QConsTarget_k$ for the groups of space types $k \varepsilon K$, as follows:

$$CostFact_{h,k,n+1} = CostFact_{h,k,n} - \{ (1 / (\lambda_{s,t,h} \cdot TrCostsSRep_{h,k,n} \cdot CostFact_{h,k,n})) \cdot AdjDamp \cdot \ln [(N \cdot RateTarget_k - n \cdot Rate_{k,n}) / (N-n) \cdot Rate_{k,n}] \} \quad (114)$$

where:

$\lambda_{s,t,h}$ = utility function dispersion parameter for development event choice between Gk and E0 for space type h (??JEA??any other ideas??)

$TrCostsRep_{h,k,n}$ = the representative average unfactored costs for all potential transitions to updated space type h per unit of space across the set of parcels considered in the n batches processed up to this point in the current run of the SD Module; calculated considering all the $TrCostsS_{v,h}$ for all the parcels processed; (??JEA??any other ideas??)

AdjDamp = cost adjustment damping factor for modifying the rate of adjustment to the cost adjustment factor with each additional batch of parcels considered;

$RateTarget_k$ = rate of space transitions required over the full set of parcels in order to match the target quantity of space construction $QConsTarget_k$ for the group of space types k

$Rate_{k,n}$ = rate of space transitions obtained for the group of space types k for the n batches of parcels processed up to this point in the current run of the SD Module

with:

$RateTarget_k = QConsTarget_k / N$

and

$Rate_{k,n} = QConsGot_{k,n} / n$

where:

$QConsGot_{k,n}$ = quantity of space construction obtained for the group of space types k for all the n batches of parcels processed up to this point in the current run of the SD Module

$RateTarget_k$ is calculated once for each group of space types k at the start of the processing of the parcels for a run of the SD Module. $TrCostsRep_{h,k,n}$ is updated as each parcel is considered. $Rate_{k,n}$ is re-calculated after each batch of parcels n is considered. The re-calculated values for $TrCostsRep_{h,k,n}$ and $Rate_{k,n}$ are used in the updating of $CostFact_{h,k,n}$ to $CostFact_{h,k,n+1}$.

The quantity of space construction obtained for the group of space types k after n batches, $QConsGot_{k,n}$ must be expressed in the same units of physical space area (such as ft^2 or m^2) as the corresponding target quantity of space $QConsTarget_k$, combining the quantities across the relevant set of space types considered in the SD Module as follows:

$$QConsGot_{k,n} = \Sigma_{h \varepsilon H(k)} UAreaC_{h,n} \cdot HFact_{h,k} \quad \text{for each } k \varepsilon K \quad (115)$$

where:

$H(k)$ = set of space types in the SD Module that are in the group of space types k related to a specific target quantity of space construction;

$UAreaC_{h,n}$ = updated quantity of space type h obtained after considering all the parcels in the n batches of parcels processed up to this point in the current run of the SD Module

$HFact_{h,k}$ = factor converting quantity in units for space type h into quantity in units of space used for target quantity of construction for group of space types k

It is the quantity of space construction obtained for the group of space types k after all N batches of parcels have been considered, $QConsGot_{k,n=N}$ that corresponds to the target quantity of space $QConsTarget_k$.

The factored adjustments to costs made after each batch of parcels is considered are designed to bring the anticipated results from the SD Module closer to the relevant targets at the end of the processing of the full set of parcels. The value for AdjDamp acts to dampen these adjustments, allowing for a model set-up where the targets may not be matched exactly. A large N (say 50 or more) and a value of 1.0 for AdjDamp should provide results that match the targets fairly closely, consistent with a stronger reliance on the equilibrium-based results obtained with the AA Module. Smaller values for N and for AdjDamp in particular will provide weaker tendencies for the results to follow the targets from the AA Module, consistent with a reduced reliance on the equilibrium results obtained with the AA Module.

At the start of a run of the SD Module, for the first batch of parcels, where n=1, the construction cost adjustment factors $CostFact_{h,k,n=1}$ for all the $h \varepsilon H(k)$ and $k \varepsilon K$ are set to the values established at the end of the previous run of the SD Module, for the previous transition from one year to the next. For the first year, these factors are all set to 1.0 to begin the SD Module processing.

At the end of a run of the SD Module, the ratio of the obtained quantity over the target quantity of space construction for a given group of space types k compares the amounts of construction activity determined in the SD Module and AA Modules for the group of

space types. To the extent that the SD Module is seen to provide a more precise representation, in that it takes into account land and zoning capacities and specific costs and rents at the parcel-level, the value determined for a given run of the SD Module for the transition from one year to the next can be used to inform the amount of considered in the AA Module for the next year. This provides a feedback from the SD Module to the AA Module, where the results from the SD Module can inform the inputs to the AA Module. In particular, the model-wide quantities of the activity or activities in the AA Module making the puts contributing to the quantity of space target $QConsTarget_k$ input to the AA Module in year t+1 could be factored in some way by the ratio of $QConsGot_{k,n=N}$ over $QConsTarget_k$ for the year t and the transition from year t to year t+1. There is no specific facility included in the PECAS software to code this sort of factoring directly. It would have to be implemented as part of the determination of the inputs to the AA Module for each year.

6.9. Derivation of Developer Choice Models Using Random Utility Theory

The two levels of developer choice model equations used in the SD Module can be derived from a single utility function using random utility theory. This derivation is outlined here. For a more complete treatment, see (Hunt et al., 2007)

Joint Choice

Consider the option for a developer to change the development state of a quantity of land to contain quantity *j* of space type *h*. The utility of the state change option is

$$RU_{hjp} = T_{hjp} j + lTr_{hjp} + l\varepsilon_s + l\varepsilon_q \quad (116)$$

where:

- T_{hjp} = net expected revenue per unit of space, accounting for rent revenue and maintenance and amortized construction costs, constant for a given *p*, *h* and within a range of *j*
- l = size of the land under consideration
- Tr_{hjp} = negative of amortised transition cost per unit of land, constant for a given *p*, *h* and within a range of *j*
- ε_s = random component of utility associated with space type alternatives S (b $\in$ S, h $\in$ S).
- ε_q = random component of utility associated with continuous quantity alternatives in the interval $\left[Q_{hp}^{\min}, Q_{hp}^{\max}\right]$

Quantity Choice

Consider the probability of choosing future quantity of development *j** given the choice of future state *h.* This is the probability that a particular quantity will have a higher utility than any of the other alternatives:

$$\begin{aligned}\Pr(j^*) \mid h &= \Pr\left(RU_{hj^*p} \geq RU_{hjp}\right) \forall Q_{hp}^{\min} < j < Q_{hp}^{\max}, j \neq j^* \\ &= \Pr\left(T_{hj^*p} j^* + lTr_{hj^*p} + l\varepsilon_s + l\varepsilon_q \geq T_{hjp} j^* + lTr_{hjp} + l\varepsilon_s + l\varepsilon_q\right) \forall Q_{hp}^{\min} < j < Q_{hp}^{\max}, j \neq j^* \quad (117)\end{aligned}$$

ε_s is constant given *h.* Assume that ε_q is Gumbel distributed with location parameter 0 and scale parameter μ_q, then the probability density function for quantity j is

$$PDF(j^*) = \frac{e^{\mu_q\left(T_{hj^*p}\frac{j^*}{l}+Tr_{hj^*p}\right)}}{\int_{j=Q_{hp}^{\min}}^{Q_{hp}^{\max}} e^{\mu_q\left(T_{hj^*p}\frac{j}{l}+Tr_{hjp}\right)}dj} \tag{118}$$

$$PDF(j^*) = \frac{e^{\mu_q\left(T_{hj^*p}\frac{j^*}{l}+Tr_{hj^*p}\right)}}{\left.\frac{le^{\mu_q\left(T_{hjp}\frac{j}{l}+Tr_{hjp}\right)}}{\mu_q T_{hjp}}\right|_{j=Q_{hp}^{\min}}^{Q_{hp}^{\max}}} \tag{119}$$

Since Tr_{hjp} and T_{hjp} are not, in general, constant across the range $Q_{hp}^{\min} < j < Q_{hp}^{\max}$ the sum across the range in the denominator will have to be performed in sections. To select a quantity *j** from this probability density function given a random draw *x* in the interval [0,1] we select

$$\int_{Q_{hp}^{\min}}^{j^*} PDF(j)dj = x \tag{120}$$

or

$$j^* = \left(\frac{\ln \frac{\mu_q T_{hjp} \left(xD_{Q_{hjp}^{\max}} - D_{Q_b} + \frac{le^{\mu_q \left(T_{hQ_b^+ p} \frac{Q_b}{l} + Tr_{hQ_b^+ p} \right)}}{\mu_q T_{hQ_b^+ p}} \right)}{l}}{\mu_q} - Tr_{hQ_b^+ p} \right) \frac{l}{T_{hQ_b^+ p}} \quad (121)$$

with

$$D_x = \left. \frac{le^{\mu_q \left(T_{hjp} \frac{j}{l} + Tr_{hjp} \right)}}{\mu_q T_{hjp}} \right|_{j=Q_{hp}^{\min}}^{x} \quad (122)$$

where:

Q_b = the highest boundary point defined by the discontinuities in Tr_{hjp} and T_{hjp} for which $D_{Q_b} < xD_{Q_{hp}^{\max}}$

Thus we have an operational continuous logit model for the development intensity simulation. A random number x is drawn and D_{Q_b} is evaluated at successively lower boundary points in the intensity range allowed by zoning regulations. j^* is calculated at the first valid boundary; being the first boundary for which $D_{Q_b} < xD_{Q_{hp}^{\max}}$.

Development State Choice

Consider the probability of a future development state h having a higher utility over any other development state h'.

$$\Pr(h) = \Pr\left(RU_{hjbvp} \geq RU_{h'jbvp}\right) \forall h' \in S, h' \neq h$$

$$= \Pr\left(\max_j \left(T_{hjp} j + lTr_{hjp} + l\varepsilon_s + l\varepsilon_q \right) \geq \right.$$

$$\max_j\left(T_{h'jp}j + lTr_{h'jp} + l\varepsilon_s + l\varepsilon_q\right)\right) \forall h' \in S, h' \neq h \tag{123}$$

Define

$$\widetilde{V}_h^r \quad = \quad l\varepsilon_s + \max_j\left(T_{hjp}j + lTr_{hjp} + l\varepsilon_q\right) \tag{124}$$

It is assumed that a smaller range of density options does not decrease the attractiveness of development state proportionally. That is, it is assumed that allowing half the density range does not cut in half the attractiveness of the development state, unless the trimming of the range removes the more desirable alternatives. Then, from continuous logit theory:

$$\widetilde{V}_h^r = l\varepsilon_s + l\left(\frac{1}{\mu_q}\ln\frac{\int_{j=Q_{hp}^{\min}}^{Q_{hp}^{\max}} e^{\mu_q\left(T_{hjp}\frac{j}{l}+Tr_{hjp}\right)}dj}{\left(Q_{hp}^{\max}-Q_{hp}^{\min}\right)} + \varepsilon_q'\right) \tag{125}$$

$$\widetilde{V}_h^r = \frac{l}{\mu_q}\ln\left(\frac{le^{\mu_q\left(T_{hjp}\frac{j}{l}+Tr_{hjp}\right)}}{\mu_q T_{hjp}}\Bigg|_{j=Q_{hp}^{\min}}^{Q_{hp}^{\max}}\right) - \frac{1}{\mu_q}\ln\left(Q_{hp}^{\max}-Q_{hp}^{\min}\right) + l\varepsilon_s + l\varepsilon_q' \tag{126}$$

with ε_q' following the same distribution as ε_q.

Again the sum across the range will have to be performed in sections if Tr_{hjp} and T_{hjp} take on different values within the range.

We choose a Gumbel distribution for $\varepsilon_s + \varepsilon'_q$, with scale parameter μ_s, again requiring a greater variance on the sum than on the second term, as follows:

$$\mu_s < \mu_q$$

This gives

$$\Pr(h) = \frac{\exp\left(\frac{\widetilde{V}_h}{l}\right)}{\sum_{h' \in S} \exp\left(\frac{\widetilde{V}_{h'}}{l}\right)} \tag{127}$$

with

$$\widetilde{V}_h = \frac{l}{\mu_q} \ln\left(\frac{l e^{\mu_q \left(T_{hjp} \frac{j}{l} + Tr_{hjp} \right)}}{\mu_q T_{hjp}} \Bigg|_{j=Q_{hp}^{\min}}^{Q_{hp}^{\max}} \right) - \frac{1}{\mu_q} \ln\left(Q_{hp}^{\max} - Q_{hp}^{\min} \right) \tag{128}$$

Thus with a piecewise constant per space unit and per land unit utility function, the utility of choosing a space type can be calculated as the expected maximum utility of choosing one particular development intensity from within a range of continuous allowable development intensities, and from this follows the probability of choosing a space type.

7. Space Development Module – Aggregate Version

7.1. Approach

An aggregate version of the Space Development Module (SD-A Module) is included as an alternative to the disaggregate version described above. It uses direct representations of aggregate quantities of developed space and land in zones. These aggregate quantities are proportioned among the possible outcomes of developer actions from one year to the next using logit-style aggregate allocation equations similar in essence to the equations used in the AA Module.

The SD-A Module, with its aggregate approach, does not provide the same level of detailed representation provided by the disaggregate version. But it does avoid some of the practical challenges that can arise with the disaggregate version related to both (a) requirements for potentially very large volumes of very fine-level data and (b) unacceptably large microsimulation error. The SD-A Module can offer a practical alternative to the SD Module in some situations, either as an interim step on the way to the implementation of an SD Module or as an alternative for specific policy analysis.

Each land use zone (LUZ) has a total quantity of land measured in area units, such as acres or hectares. Categories of land are defined, and each category has some portion of the total quantity of land in the zone along with a specific set of zoning rules that dictate what kinds of development actions are permitted on this category of land.

Quantities of developed space (called ‘space’) of different types in the zone are also measured in areas, such as square feet or square meters. These quantities of space are situated on the land in the zone. The space is said to ‘use’ or ‘occupy’ the land. Space of a given type is a put consumed by activities in the AA Module. It is ‘non-transportable’ in that it must be consumed by the activity in the zone where it is located. At any point in the model run, the aggregate quantity of space of a given type in a given zone is the sum of the quantities of that space on all of the categories of land in the zone.

The zoning rules for each category of land in each zone are specified by the analyst as part of a run of the SD-A Module. These rules list the types of space that can be developed and the maximum densities allowed for this development.

Thus, for each zone there are quantities of land of different categories, and each such quantity of land contains existing quantities of space of different types along with quantities of unused capacity for additional space of different types. The quantity of unused capacity for additional space of a given type on the land of a given category is the difference between the quantity at the maximum density specified in the zoning rules and the current quantity. In general, there will be zones containing land of only a subset of the full set of categories considered in the model – that is, where the quantity of land is 0 of some categories. Again, in general, there will be categories of land where some or even all of the space types cannot be developed according to the zoning rules. Further, in some cases, the existing quantity of space of a specific type on the land of a specific category may be 0; or the quantity of unused capacity for additional space of a specific type on the land of a specific category may be 0. These relationships among quantities of land by category, existing quantities of space by type and quantities of unused capacity for additional space by type are illustrated in Figure 09.

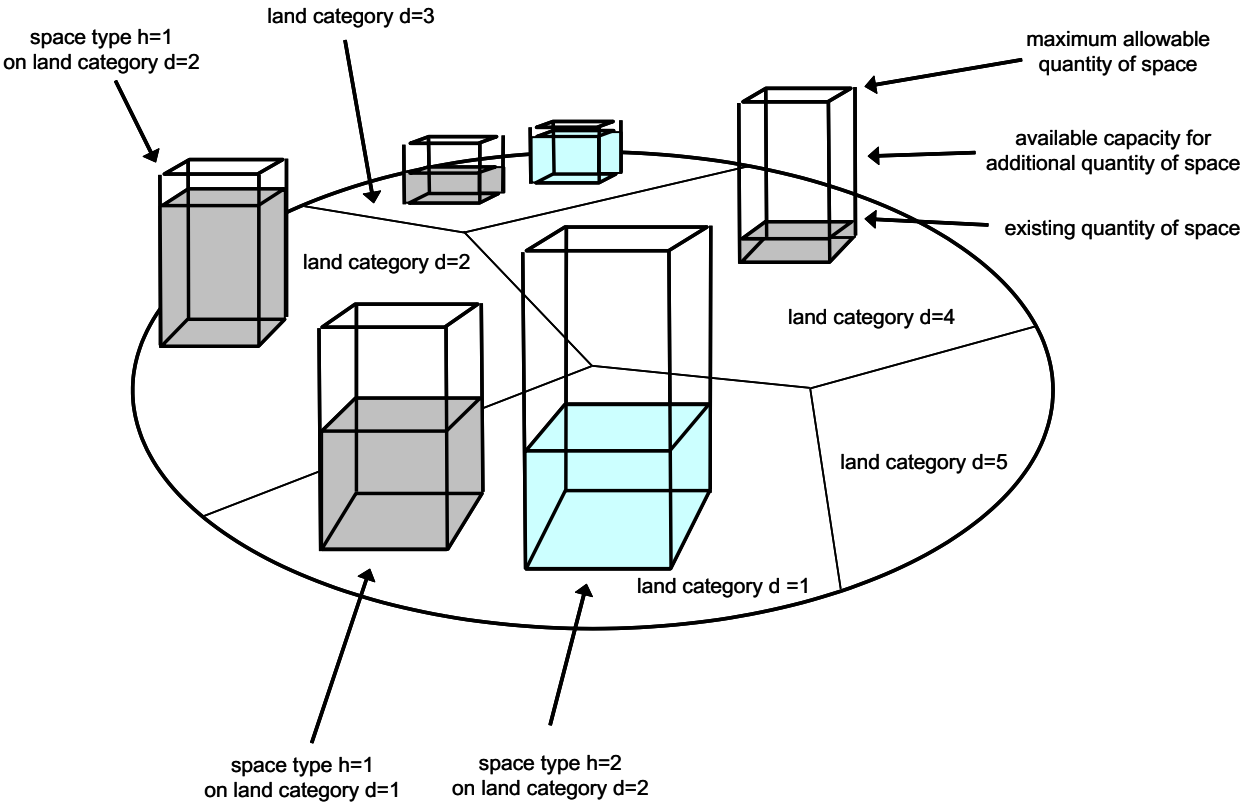

Figure 09: Quantities of existing space and available capacity for additional space by type in land by category in a single land use zone.

In a given year-to-year step considered with the SD-A Module, the aggregate results of developer actions (demolishing, building and transforming space) in each zone are simulated by updating the existing quantities of space by type and the available capacities for additional space by type in the zone. This is done in two sets of allocations, the first considering what happens to the existing space in the step and the second considering what happens to the available capacities in the step. These two sets of allocations take into account the zoning rules that apply and the extent of enforcement of these rules, along with certain (aggregate-level) representations of the net financial returns to developers, including in particular:

- The existing quantities of space in the zone;

- The zoning rules specifying the types of space that are permitted and the maximum densities at which they are permitted for each category of land in the zone;
- The further options available for existing space;
- The (typical) unit costs and fees associated with development of each permitted space type; and
- The price for each type of permitted space in the zone and in the wider model area.

The first set of allocations considers the existing quantities of space. A separate allocation is done for each quantity of existing space on a given category of land in a given zone. In this separate allocation, the quantity of existing space is allocated among the set of available space types.

These allocations concerning existing quantities of space are done in a series of iterations using an approach similar to that used in some capacity-restrained traffic assignment techniques – in order to reduce the extent of any interim quantities exceeding capacities. In each such iteration, proportions of the total quantities of existing space by type are allocated using current space quantities. These current space quantities are updated in each iteration to incorporate the results of the allocations in previous iterations, so that the allocations can respond more precisely as certain quantities approach the relevant capacities. These iterations are continued until all the existing space has been allocated.

The set of available space types in these allocations concerning existing space includes:

(a) ‘permitted’ space types that can be developed as new space according to the zoning rules;

(b) ‘recognized’ space types that appear as new space even through they are not allowed according to the zoning rules; and

(c) ‘admitted’ space types reflecting further options available for existing space.

The space types included in these three categories are specified separately in the zoning rules for each land category and zone, and thus can vary across both land categories and zones.

The ‘permitted’ space types can include any one or more of those represented in the model, or none of them (as would be the case for lands where no development of any kind is allowed).

The ‘admitted’ space types can include:

- the existing type, recognizing the ‘no change’ option that is typically available for existing space regardless of the types included or excluded in the set of ‘permitted’ space types; this provides representation of ‘grandfathering’ where existing space not conforming to (perhaps updated) zoning rules is allowed to remain even though new space of this type is not allowed. Further, in the inputs to the software this ‘grandfathering’ is indicated to be available or not for each land category for each zone.

- derelict space, as a specific type of space; including this alternative provides representation of the potential for landlords to neglect or abandon quantities of existing space to the point where the space is no longer usable as a put of its existing type, and is derelict;

The ‘recognized’ space types can include:

- informal forms of space that are not recognized directly in the zoning rules but nevertheless are appearing in sufficient quantities in reality to warrant that they are included as alternatives in the model, such as slums as a type of residential space; and

- space types that are allowed in some land categories and zones but not others, according to the zoning rules, and yet are appearing where they are not allowed in sufficient quantities in reality that it is appropriate to include their potential in the model, providing representation of rogue development and the results of lax enforcement.

After the existing space is allocated, the resulting interim quantities of space by type are calculated. The interim quantity of a given type of space (on a category of land in a zone) is the sum of the quantities allocated to that type in all the allocations done for all the existing quantities of space (again, on that category of land in that zone).

The second set of allocations considers the available capacities. A separate allocation is done for each quantity of available capacity for a type of space on a category of land in a zone. The types of space considered are those included in the 'permitted' and 'recognized' categories identified above. The 'admitted' category concerns existing space and thus does not apply. First, the quantities of available capacity are calculated. Then, each of these quantities is allocated to either (a) new space of that type or (b) retained capacity available for subsequent development. The quantity of available capacity is calculated by subtracting the interim quantity of the space (on the land in the zone, determined after the first set of allocations as described above) from the maximum allowable quantity of the space according to the specified maximum density in the zoning rules and the relevant quantity of land.

The quantities of new space established in the second set of allocations are added to the interim quantities determined after the first set of allocations in order to calculate the updated quantities of space for the year-to-year step. These updated quantities are used in the AA Module for year t+1.

7.2. Typical Categories in Application

A typical set of space types for an implementation of the SD-A Module is as follows:

- Low Density Housing space
- Medium Density Housing space
- High Density Housing space
- Industrial space
- Warehouse space
- Retail space
- Office Grade A space
- Office Grade B space
- Hospital space
- Gradeschool space
- Post-Secondary Institutional space
- Government space
- Agricultural space
- Derelict space.

These must correspond to the space types included as individual put categories in the AA Module. The quantities for these types must all be expressed in the same units, for example all in square feet or all in square meters. In general, standard floorspace categories in land use descriptions are used as the starting point for developing the set of space types.

Note that this list of typical space types is the same as the list for the disaggregate SD Module including in Section 6.2 above. Any space type considered in the SD Module can

also be considered in the SD-A Module, consistent with the ability to use either the SD Module or the SD-A Module with a given AA Module.

A typical set of land categories for an implementation of the SD-A Module is as follows:

- Low Density Residential land
- Medium Density Residential land
- High Density Residential land
- Industrial land
- Commercial land
- Gradeschool land
- Mixed Use land
- Institutional land
- Government land
- Agricultural land
- Park And Green land
- Development Moderately Restricted land
- Development Heavily Restricted land.

In general, standard zoning categories in municipal plans and by-laws are used as the starting point for developing the set of land categories. The quantities for these categories must all be expressed in the same units, for example all in acres or all in hectares.

The total quantity of land in a zone is fixed for a particular model, for the entire model run and for all model runs considering different scenarios. The quantities of land in the different categories can change, and in general will change with changes in the land use policy inputs to the model, but the sum of the quantities in the zone is fixed.

The zoning rules for each land use zone (LUZ) are specified in a table format indicating the permitted and recognized space types and associated maximum densities along with the admitted space types for each land category in the zone. Figure 10 shows an example of such a zoning rules table for a zone in an implementation of the SD-A Module with the land categories and space types listed above.

There is one such table specified for each zone for each year-to-year step considered in the model.

A maximum density is specified for each space type in each case. If in reality the zoning rules specify a joint maximum density for a range of potential space types in combination, this joint maximum density must be allocated among the individual potential space types a priori and outside the model and the resulting separate maximum density values for the potential space types input to the model.

space types — full range

land categories — full range

	No Change	Derelict	Low Density Housing	Medium Density Housing	High Density Housing	Industrial	Warehouse	Retail	Office Grade A	Office Grade B	Hospital	Gradeschool	Post-Secondary Institutional	Government	Agricultural
Low Density Residential	a	a	p 0.02												
Medium Density Residential	a	a		p 1.50											
High Density Residential	a			r 1.50	p 15.00				r 10.00						
Industrial	a	a		r 1.50		p 0.50	p 1.00	r 1.50		r 15.00		r 0.05	r 0.10		r 1.00
Commercial	a	a	r 0.10	p 1.50	p 15.00	r 0.75	p 1.00	p 2.00	p 15.00	p 15.00	r 5.00	r 0.05	r 0.10		r 1.00
Gradeschool	a											p 0.05		r 5.00	
Mixed Use	a	a		p 1.50	p 15.00		p 1.00	p 2.00	p 15.00						
Institutional	a												p 0.10	p 0.10	
Government	a							r 0.05	p 5.00					p 2.00	
Agricultural	a														p 1.00
Park And Green	a														
Development Moderately Restricted	a	a	p 0.01										p 0.01	r 0.05	p 1.00
Development Severely Restricted	a												p 0.01		r 0.01

Key: p / 10.00 = space type category; a=admitted, p=permitted, r=recognized; maximum permitted density, $MaxDen_{l,d}$

Figure 10: Example of zoning rules table for a particular zone with typical space types and land categories.

7.3. Mathematical Formulation

In a given year-to-year step, going from the pervious year to the next year, the price for each type of space in each LUZ determined in the AA Module for the previous year is input to the SD-A Module. Figure 1 shows this 'space prices' information flow. The SD-A Module then starts the process of working through the LUZ, performing the allocations for each zone in turn, and thereby determining the updated (year t+1) quantities of space of each type in the zone given the existing (year t) quantities of space, available capacities, and prices. These allocations and the equations used in them are described below.

Existing Space Allocation

In the first set of allocations, the SD-A Module allocates the quantities of existing space on each land category to interim space types. Each quantity of existing space on a given category of land is allocated among the available space types using a logit function as follows:

$$IntSQty_{v,i,d} = (1/ExSits) \cdot ExtSQty_{v,d} \cdot (\exp(\lambda_{i,d} \cdot UIntS_{v,i,d}) / \Sigma_{i \in I(d)} (\exp(\lambda_{i,d} \cdot UIntS_{v,i,d})))$$

(129)

with:

$$UIntS_{v,i,d} = \alpha_{price,i} \cdot (Price_i - StdExPr_i) + \alpha_{maprice,i} \cdot (MAPrice_i - StdExMAPr_i) + \alpha_{cap,i} \cdot (1 + ApFac_d \cdot (CurSQty_{i,d} / CapS_{i,d})^{\omega_d}) + \alpha_{exprop,i} \cdot IntSProp_{i,d} + ATrCon_{v,i} + RTrCon_{v,i} + NCCon_i \quad (130)$$

with:

$$IntSProp_{i,d} = ExtSQty_{v=i,d} / \Sigma_{i \in I(d)} ExtSQty_{i,d} \quad (131)$$

$$CapS_{i,d} = LandSize_d \cdot MaxDen_{i,d} \quad (132)$$

where:

d = index representing land categories;
v = index representing existing space types;
$V(d)$ = set of existing space types on land category d in the zone;
i = index representing interim space types;
$I(d)$ = set of available interim space types, including all space types in the 'permitted', 'admitted' and 'recognized' categories on land category d in the zone;
$UIntS_{v,i,d}$ = utility for a unit of existing space type v on land category d being allocated to interim space type i in the zone;
$Price_i$ = price (rent) for interim space type i in zone determined in AA Module for current year; in units of money per unit of area for space type i per year;
$StdExPr_i$ = standard reference price per unit for transforming existing space to interim space type i ;
$MAPrice_i$ = model-wide average price (rent) for interim space type i determined in AA Module for current year; in units of money per unit of area for space type i per year;
$StdExMAPr_i$ = standard reference model-wide average price per unit for transforming existing space to interim space type i;
$ApFac_d$ = multiplicative parameter for capacity term in utility function for allocating existing space on land category d;
$CurSQty_{i,d}$ = current quantity of space type i on land category d in the zone;
$CapS_{i,d}$ = quantity of available capacity for interim space type i on land category d in the zone;
$LandSize_d$ = quantity of land of category d in the zone;
$MaxDen_{i,d}$ = specified maximum allowable density for space type i on land category d in the zone; expressed in units of space per unit of land;
$\omega_{i,d}$ = exponent parameter for capacity term in utility function for allocating existing space on land category d;
$\lambda_{i,d}$ = utility function dispersion parameter for allocation of existing space to interim space types on land category d;
$IntSProp_{i,d}$ = proportion of existing space of type i on land category d in the zone;
$ATrCon_{v,i}$ = utility function constant for allocation of existing space type v to interim space type i; this has a non-zero value and thus applies only when the interim space type i is in the 'permitted' category;
$RTrCon_{v,i}$ = utility function constant for allocation of existing space type v to interim space type i; this has a non-zero value and thus applies only when the interim space type i is in the 'recognized' category;
$NCCon_i$ = utility function constant for allocation of existing space to the no change option; this has a non-zero value and thus applies only for the no change option;
$\alpha_{price,i}$ = utility function coefficient for the sensitivity to price when allocating to interim space types;

$\alpha_{maprice,i}$ = utility function coefficient for the sensitivity to model-wide average price when allocating to interim space types;
$\alpha_{cap,i}$ = utility function coefficient for the sensitivity to available capacity when allocating to interim space types;
$\alpha_{exprop,i}$ = utility function coefficient for the sensitivity to existing proportions of space by type when allocating to interim space types;

The allocations using the above equations are performed for the full set of existing space type on a given land category in a series of iterations. The number of these iterations, denoted ExSits, is an integer greater than 0 that is specified as part of the model inputs and is constant across all land categories and zones. In each such iteration, the proportion 1/ExSits of the quantity of each of the existing space types is allocated to the interim space types using the above equation, with the current quantities of space by type (needed to establish the available capacities) first updated for use in the iteration. The calculation of the current quantities of space by type for use in each of these iterations depends on the iteration.

In the first iteration, the current quantity of space of each type is the existing quantity, as follows:

$$CurSQty_{i,d} = ExtSQty_{v=i,d} \quad \text{for each i} \tag{133}$$

In each subsequent iteration, the current quantity of space of each type is its current quantity in the previous iteration minus the portion of its existing quantity allocated in that previous iteration plus the quantities allocated to it in the previous iteration, as follows:

$$CurSQty_{i,d} = PrevSQty_{i,d} - (1/ExSits) \cdot ExtSQty_{v=i,d} + \Sigma_{v \in V(d)} IntSQty_{v,i,d} \quad \text{for each i} \tag{134}$$

where:

$PrevSQty_{i,d}$ = current quantity of space type i on land category d in the previous iteration in the sequence of iterations allocating proportions of existing space to interim space types;

For space types in the 'permitted' category, the constants $RTrCon_{v,i}$ and $NCCon_i$ do not apply and are fixed at 0.

For space types in the 'admitted' category, the capacity term in the utility is not relevant and therefore it is not calculated and the value for $\alpha_{cap,i}$ is fixed at 0. The constants $RTrCon_{v,i}$ also do not apply and are fixed at 0. The constant $NCCon_i$ has a non-zero value only for the no change option within the 'admitted' category.

For space types in the 'recognized' category, the constants $RTrCon_{v,i}$ and $NCCon_i$ do not apply and are fixed at 0.

With the different sets of constants applying for the 'permitted' and 'recognized' categories, the model can be calibrated to reflect differing tendencies for space development inside and outside of the strict zoning rules as appropriate. Note that any corresponding tendencies for space quantities to exceed those specified in the strict zoning rules can be reflected in the function parameters related to the capacity term in the utility function and in higher maximum densities if appropriate.

In many cases, the existing space type is included in both the 'permitted' category (as an option for new space development) and the 'admitted' category (as the 'no change' option). In such cases, this space type is available in two places in each allocation, once for each of these two options. The utility value and resulting allocation for each option is established in the same way it would be otherwise as described above. But the same value for the interim space type, i=v, is used for both options, and the current value for the space type is calculated twice in each iteration, both times using Equation 123, once as the new space amount and once as the no change amount.

This set of ExSits iterations is done for all categories of land in each zone.

Interim Space Quantities

The results of the existing space allocations are summed to establish the total interim space quantities. In each zone, the interim quantity of space type i on land category d is the sum of all the existing amounts allocated to it, which is established by updating its current quantity to include the results of the last (and thus final 'previous') iteration of allocations as follows:

$$IntSVQty_{i,d} = PrevSQty_{i,d} - (1/ExSits)\cdot ExtSQty_{v=i,d} + \Sigma_{v\in V(d)}\, IntSQty_{v,i,d} \quad \text{for each } i\varepsilon I(d) \quad (135)$$

where:

$IntSVQty_{i,d}$ = interim quantity of space type i on land category d in the zone;

Figure 11 depicts the process of allocating existing space to interim space types and the calculation of interim space total quantities of interim space.

allocating to interim space

allocating from existing space	Row Totals	Type 1	Type 2	Type 3	Type 4
Type 1	6475	4820 / 25	280	130	1220
Type 2	13190	830	11750 / 80	220	310
Type 3	3490	480	160	2720 / 50	80
Type 4	6210	520	300	180	5190 / 20
Column Totals	29365	6675	12570	3300	6820

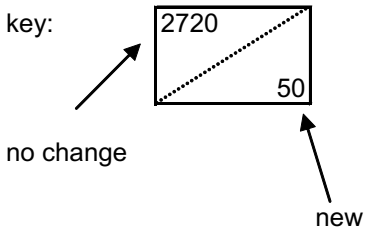

Figure 11: Allocating existing space quantities to interim space quantities. Each existing space quantity total in the left-most column is allocated among the interim types in the other columns to the right. Then the interim space quantity totals are obtained by summing down the columns.

Available Capacity Quantities

The quantity of available capacity for each 'permitted' and 'recognized' space type to be considered in the second set of allocations is calculated by subtracting the interim quantity of the space from the maximum allowable quantity of the space according to the specified maximum density in the zoning rules and the relevant quantity of land, as follows:

$$AvCapS_{a,d} = (LandSize_d \cdot MaxDen_{a,d}) - IntSVQty_{i=a,d} \quad \text{for each } a \varepsilon A(d) \qquad (136)$$

where:

a = index representing space types that can have available capacity according to the zoning rules (that are in the 'permitted' or 'recognized' categories);
A(d) = set of space types that can have available capacity according to the zoning rules (that are in the 'permitted' or 'recognized' categories) on land category d in the zone;
$AvCapS_{a,d}$ = available capacity for space type a on land category d in the zone;

Available Capacity Allocation

In the second set of allocations, the SD-A Module allocates the available capacities for development to either new space or retained capacity. Each quantity of available capacity for a given space type on a given land category is allocated among these two options using a logit function as follows:

$$NuSQty_{a,d} = AvCapS_{a,d} \cdot \{ \exp(\lambda_{a,d} \cdot UNuS_{a,d}) / (1 + \exp(\lambda_{a,d} \cdot UNuS_{a,d})) \} \qquad (137)$$

with:

$$UNuS_{a,d} = \alpha_{price,a} \cdot (Price_a - StdNuPr_a) + \alpha_{maprice,a} \cdot (MAPrice_a - StdNuMAPr_a) + \alpha_{exprop,a} \cdot NuProp_{a,d} + ANuCon_a + RNuCon_a \qquad (138)$$

with:

$$NuProp_{a,d} = IntSVQty_{i=a,d} / (LandSize_d \cdot MaxDen_{a,d}) \qquad (139)$$

where:

$NuSQty_{a,d}$ = quantity of new space of type a on land category d arising with the use of the relevant available capacity for space type a on land category d in the zone;
$UNuS_{a,d}$ = utility for a unit of available capacity for space type a on land category d being allocated to new space in the zone;
$Price_a$ = price (rent) for space type a in zone determined in AA Module for current year; in units of money per unit of area for space per year;
$StdNuPr_a$ = standard reference price per unit for transforming relevant available capacity into new space for space type a;
$MAPrice_a$ = model-wide average price (rent) for space type a determined in AA Module for current year; in units of money per unit of area for space per year;
$StdNuMAPr_a$ = standard reference model-wide average price per unit for transforming relevant available capacity into new space for space type a;
$\lambda_{a,d}$ = utility function dispersion parameter for allocation of available capacity on land category d;
$NuProp_{a,d}$ = proportion of full capacity for space type a already used to contain space on land category d in the zone;
$ANuCon_a$ = utility function constant for allocation of available capacity to new space for space type a; this has a non-zero value and thus applies only when the new space type a is in the 'permitted' category;
$RNuCon_a$ = utility function constant for allocation of available capacity to new space for space type a; this has a non-zero value and thus applies only when the new space type a is in the 'recognized' category;
$\alpha_{price,a}$ = utility function coefficient for the sensitivity to price when allocating available capacity;
$\alpha_{maprice,a}$ = utility function coefficient for the sensitivity to model-wide average price when allocating available capacity;
$\alpha_{exprop,a}$ = utility function coefficient for the sensitivity to existing proportions of space by type when allocating available capacity.

Updated Space Quantities

The new space quantities established in the second set of allocations are added to the total interim space quantities to establish the updated space quantities. In each zone, the updated quantity of space type h on land category d is the sum of the corresponding interim space and new space quantities, as follows:

$$UpSQty_{h,d} = IntSVQty_{i,d} + NuSQty_{a=h,d} \quad \text{for each } h \varepsilon H(d) \qquad (140)$$

where:

$UpSQty_{h,d}$ = updated quantity of space type h on land category d in the zone;

In these summations, the value for $NuSQty_{a=h,d}$ is set to 0 when the space type h is in the 'admitted' category, not the 'permitted' or 'recognized' categories, because there is no available capacity and thus no new space in such cases.

The updated quantities of space by type for each zone overall, for the entire set of land categories for the zone, are established by summing the updated quantities over the set of land categories as follows:

$$SpcQty_{h,z} = \Sigma_{d \in D(z)} UpSQty_{h,d,z} \quad \text{for each } h \varepsilon H(d) \text{ for each d for each z} \quad (141)$$

where:

z = index representing zones;
D(z) = set of land categories in zone z;
$SpcQty_{h,z}$ = updated aggregate quantity of space type h in land use zone z;

The $SpcQty_{h,z}$ are the quantities of space used in the AA Module for the next year (t+1).

7.4. Processing Algorithm

A flowchart of the processes used in the SD-A Module is shown in Figure 12, including the model components and the information flows among them.

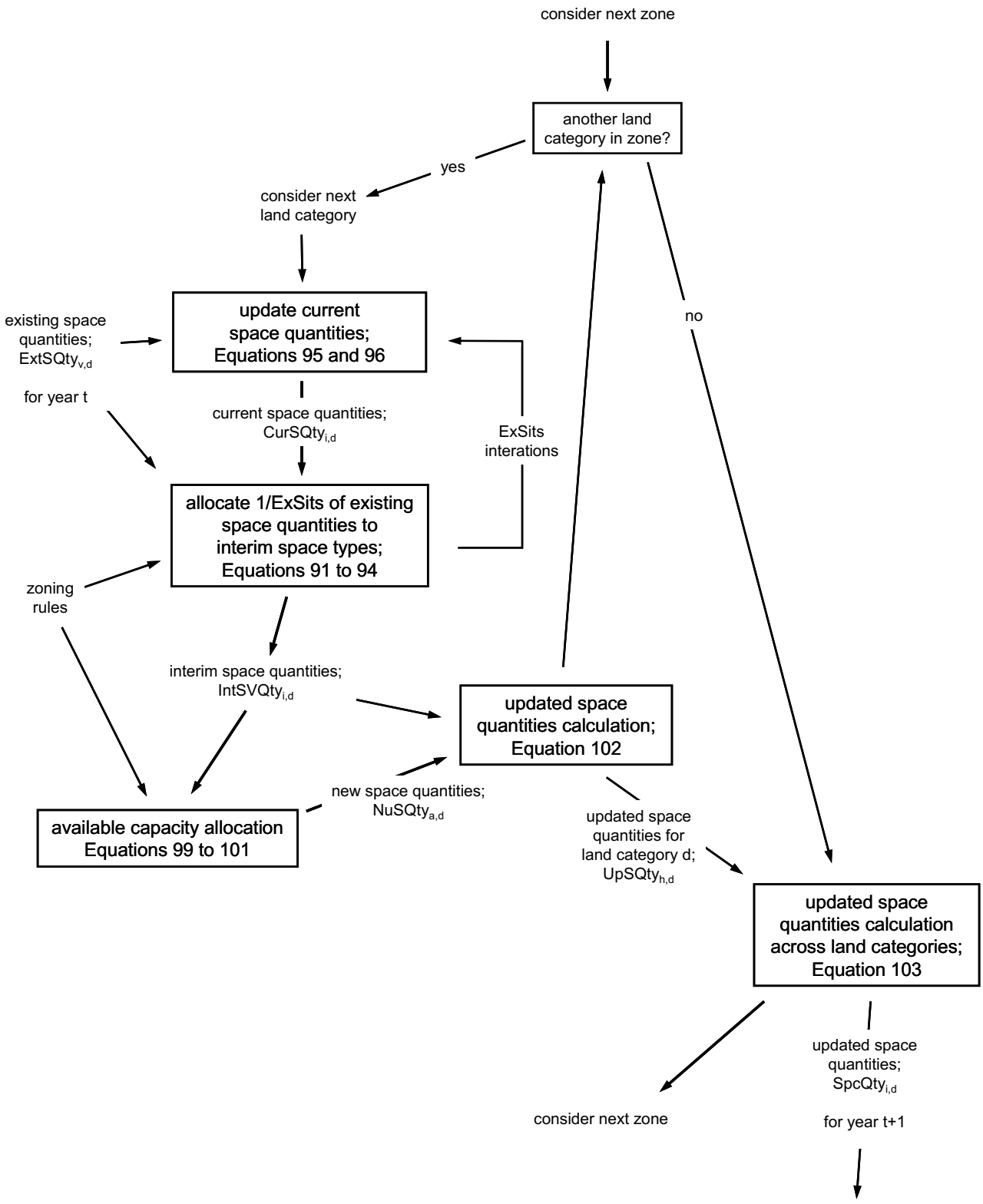

Figure 12: Steps and information flows in SD-A Module