

DAFTAR PUSTAKA

- Ahmed, B., Kamruzzaman, M. D., Zhu, X., Shahinoor Rahman, M. D., & Choi, K. (2013). Simulating Land Cover Changes and Their Impacts on Land Surface Temperature in Dhaka, Bangladesh. *Remote Sensing*, 5, 5969–5998. <https://doi.org/10.3390/rs5115969>
- Arsyad, S. (1989). *Konservasi Tanah & Air*. IPB Press.
- Azevedo, J. A., Chapman, L., & Muller, C. L. (2016). Quantifying the daytime and night-time urban heat Island in Birmingham, UK: A comparison of satellite derived land surface temperature and high resolution air temperature observations. *Remote Sensing*, 8(2). <https://doi.org/10.3390/rs8020153>
- Badan Standardisasi Nasional. (2010). Klasifikasi Penutup Lahan. In *SNI 7645:2010*.
- Baja, S. (2012). Perencanaan Tata Guna Lahan dalam Pengembangan Wilayah. In *Pendekatan Spasial & Aplikasinya*. Penerbit Andi.
- Bappeda. (2015). *Peraturan Daerah Kota Makassar Nomor 4 Tahun 2015 Tentang Rencana Tata Ruang Wilayah Kota Makassar 2015-2034*.
- Bappeda Kota Makassar. (2018). *Rencana Program Investasi Jangka Menengah (RPIJM) Bidang Cipta Karya Kota Makassar 2019-2023*.
- Bappenas. (2017). *Bunga Rampai Tesis/Disertasi (Tema: Geografi)*. Pusbindiklatren-Bappenas.
- BPS Kota Makassar. (2020). *Kota Makassar Dalam Angka Tahun 2020*. BPS Kota Makassar.
- Bureau of Environment Tokyo Metropolitan Government. (2018). *Urban Heat Island*. http://www.kankyo.metro.tokyo.jp/en/climate/heat_island.html
- Buyantuyev, A., & Wu, J. (2010). Urban Heat Islands and Landscape Heterogeneity: Linking Spatiotemporal Variations in Surface temperatures to Land-Cover and Socioeconomic Patterns. *Landscape Ecology*, 25, 17–33. <https://doi.org/10.1007/s10980-009-9402-4>
- Crago, R. D., & Qualls, R. J. (2014). Use of Land Surface Temperature to Estimate Surface Energy fluxes : Contributions of Wilfried Brutsaert and Collaborators. *Water Resources Research*, 50, 3396–3408. <https://doi.org/10.1002/2013WR015223>. Received
- Darlina, S. P., Sasmito, B., & Yuwono, B. D. (2018). Analisis Fenomena Urban Heat Island Serta Mitigasinya (Studi Kasus : Kota Semarang). *Jurnal Geodesi Undip*, 7(3), 77–87.
- Darmawan, A., Harianto, S. P., Santoso, T., & Winarno, G. D. (2018). *Buku Ajar Penginderaan Jauh Untuk Kehutanan*.
- Deep, S., & Saklani, A. (2014). Urban Sprawl Modeling Using Cellular Automata. *Egyptian Journal of Remote Sensing and Space Science*, 17, 179–187. <https://doi.org/10.1016/j.ejrs.2014.07.001>
- Deilami, K., Kamruzzaman, M., & Liu, Y. (2018). Urban Heat Island Effect: A Systematic Review of Spatio-temporal Factors, Data, Methods, and Mitigation Measures. *International Journal of Applied Earth Observation and Geoinformation*, 67, 30–42. <https://doi.org/10.1016/j.jag.2017.12.009>
- Eastman, J. R. (2012). *IDRISI Selva Tutorial*. Idrisi Production.

- [https://doi.org/10.1016/0301-0104\(80\)85118-4](https://doi.org/10.1016/0301-0104(80)85118-4)
- Eisavi, V., Homayouni, S., Yazdi, A. M., & Alimohammadi, A. (2015). Land Cover Mapping Based on Random Forest Classification of Multitemporal Spectral and Thermal Images. *Environmental Monitoring and Assessment*, 187(291), 1–14. <https://doi.org/10.1007/s10661-015-4489-3>
- El-Hattab, M., Amany, S. M., & Lamia, G. E. (2018). Monitoring and assessment of urban heat islands over the Southern region of Cairo Governorate, Egypt. *Egyptian Journal of Remote Sensing and Space Science*, 21(3), 311–323. <https://doi.org/10.1016/j.ejrs.2017.08.008>
- El Kenawy, A. M., Hereher, M., Robaa, S. M., McCabe, M. F., Moreno, J. I. L., Castro, F. D., Gaber, I. M., Al-Awadhi, T., Al-Buloshi, A., Nasiri, N. Al, Al-Hatrushi, S., Schuwerack, P. M., Angulo, D. P., Abdelaal, M. M., & Serrano, S. M. V. (2020). Nocturnal Surface Urban Heat Island Over Greater Cairo: Spatial Morphology, Temporal Trends and Links to Land-Atmosphere Influences. *Remote Sensing*, 12(3889), 1–29. <https://doi.org/10.3390/rs12233889>
- EPA. (2014). Reducing Urban Heat Islands : Compendium of Strategies. In *Urban Heat Island Basics*. Climate Protection Partnership Division Environmental Protection Agency's Office.
- Ermida, S. L., Soares, P., Mantas, V., Götsche, F. M., & Trigo, I. F. (2020). Google Earth Engine Open-Source Code for Land Surface Temperature Estimation from the Landsat Series. *Remote Sensing*, 12(1741), 1–21. <https://doi.org/10.3390/RS12091471>
- ESA. (2018). *Land Surface Temperature - Applications - Sentinel-3 SLSTR - Sentinel Online*. <https://sentinel.esa.int/web/sentinel/user-guides/sentinel-3-slstr/overview/geophysical-measurements/land-surface-temperature>
- Fadillah, D. (2019). Analisis Kerawanan Tanah Longsor Menggunakan Model Integrasi Fuzzy Logic dan Analytical Hierarchy Process (AHP). In *Skripsi*. Universitas Hasanuddin.
- FAO. (2016). *Land Cover Classification System*. <http://www.fao.org/docrep/003/x0596e/x0596e00.htm>
- Fardani, I., Mohamed, F. A. J., & Chofyan, I. (2020). Pemanfaatan Prediksi Tutupan Lahan Berbasis Cellular Automata-Markov dalam Evaluasi Rencana Tata Ruang. *MKG*, 21(2), 157–169.
- Farina, A. (2012). Exploring the Relationship Between Land Surface Temperature and Vegetation Abundance for Urban Heat Island Mitigation in Seville, Spain. In *Thesis*. Lund University.
- Firozjaei, M. K., Kiavarz, M., Alavipanah, S. K., Lakes, T., & Qureshi, S. (2018). Monitoring and Forecasting Heat Island Intensity Through Multi-Temporal Image Analysis and Cellular Automata-Markov chain Modelling: A case of Babol city, Iran. *Ecological Indicators*, 91, 155–170. <https://doi.org/10.1016/j.ecolind.2018.03.052>
- Fitriana, A., Subiyanto, S., & Firdaus, H. (2017). Model Cellular Automata Markov Untuk Prediksi Perkembangan Fisik Wilayah Permukiman Kota Surakarta Menggunakan Sistem Informasi Geografis. *Jurnal Geodesi Undip*, 6(4), 246–253.

- Gidey, E., Dikinya, O., Sebego, R., Segosebe, E., & Zenebe, A. (2017). Cellular automata and Markov Chain (CA_Markov) Model-Based Predictions of Future Land Use and Land Cover Scenarios (2015–2033) in Raya, Northern Ethiopia. *Modeling Earth Systems and Environment*. <https://doi.org/10.1007/s40808-017-0397-6>
- Haashemi, S., Weng, Q., Darvishi, A., & Alavipanah, S. K. (2016). Seasonal Variations of the Surface Urban Heat Island in a Semi-Arid City. *Remote Sensing*, 8(352), 1–17. <https://doi.org/10.3390/rs8040352>
- Hadi, B. S. (2019). *Penginderaan Jauh Digital*. UNY Press. <http://staffnew.uny.ac.id/upload/132240452/penelitian/Penginderaan Jauh Pengangtar ke Arah Pembelajaran Berpikir Spasial.pdf>
- Hakim, P. R., Rahman, A., Suhermanto, & Rachim, E. (2012). Model Koreksi Geometri Sistematik Data Imager Pushbroom Menggunakan Metode Proyeksi Kolinear. *Jurnal Teknologi Dirgantara*, 10(2), 121–132.
- Hasanlou, M., & Mostofi, N. (2015). Investigating Urban Heat Island Effects and Relation Between Various Land Cover Indices in Tehran City Using Landsat 8 Imagery. *1st International Electronic Conference on Remote Sensing*, 22 June-5 July, 1–11. <https://doi.org/10.3390/ecrs-1-f004>
- Hedge, N. P., MuraliKrishna, I. V., & ChalapatiRao, K. V. (2009). Study of Cellular Automata Models for Urban Growth. In *Map World Forum*.
- Iacono, M., Levinson, D., El-Geneidy, A., & Wasfi, R. (2015). A Markov Chain Model of Land Use Change in the Twin Cities, 1985–2005. *Tema. Journal of Land Use, Mobility and Environment*, 8(3), 263–276. <http://www.tema.unina.it/index.php/tema/article/view/2985>
- Jatmiko, R. H. (2016). Penggunaan Citra Saluran Inframerah Termal Untuk Studi Perubahan Liputan Lahan dan Suhu Sebagai Indikator Perubahan Iklim Perkotaan di Yogyakarta. In *Dissertasi*. Universitas Gadjah Mada.
- Jensen, J. R. (2015). *Introductory Digital Image Processing A Remote Sensing Perspective 4th Edition* (4th ed.). Pearson Education.
- Keeratkasikorn, C., & Bonafoni, S. (2018). Urban heat island analysis over the land use zoning plan of Bangkok by means of Landsat 8 imagery. *Remote Sensing*, 10(3). <https://doi.org/10.3390/rs10030440>
- Kurniati, R. (2019). Pengaruh Ketersediaan Ruang Terbuka Hijau Terhadap Urban Heat Island di Kota Makassar, Sulawesi Selatan. In *Thesis*. Universitas Gadjah Mada.
- Kurniati, R., & Rahmi, D. H. (2020). Ketersediaan Ruang Terbuka Hijau dan Urban Heat Island di Kota Makassar. *Jurnal Litbang Sukowati*, 3(2), 1–14.
- Kurniawan, M. I. (2016). Analisis Geospasial Terhadap Konversi Lahan Dengan Menggunakan Metode Cellular Automata. In *Skripsi*. Universitas Hasanuddin.
- Kurniawan, W. D. W. (2019). Probabilitas Perubahan Tutupan Lahan Berdasarkan Keberadaan Lokasi Wisata di Wilayah Pesisir Sarbagita. *SPACE*, 1(1), 33–39.
- Kushardono, D. (2017). *Klasifikasi Digital Pada Penginderaan jauh*. Penerbit IPB Press. <https://doi.org/10.1007/978-613-6-5006>
- Kusumadewi, S., & Guswaludin, I. (2005). Fuzzy Multi-Criteria Decision Making. *Media Informatika*, 3(1), 25–38.
- Li, H., Zhou, Y., Li, X., Meng, L., Wang, X., Wu, S., & Sodoudi, S. (2018). A new

- method to quantify surface urban heat island intensity. *Science of the Total Environment*. <https://doi.org/10.1016/j.scitotenv.2017.11.360>
- Lillesand, T. M., Kiefer, R. W., & Chipman, J. W. (2015). Remote Sensing and Image Interpretation. In *Photogrammetric Engineering & Remote Sensing* (7th ed., Vol. 81, Issue 8). John Wiley & Sons. <https://doi.org/10.14358/pers.81.8.615>
- Liu, Y. (2009). *Modelling Urban Development with Geographical Information Systems and Cellular Automata*. Taylor & Francis Group.
- Lu, L., Weng, Q., Xiao, D., Guo, H., Li, Q., & Hui, W. (2020). Spatiotemporal Variation of Surface Urban Heat Islands in Relation to Land Cover Composition and Configuration: A Multi-Scale Case Study of Xi'an, China. *Remote Sensing*, 12(2713), 1–19. <https://doi.org/10.3390/RS12172713>
- Lukiawan, R., Purwanto, E. H., & Ayundyahrini, M. (2019). Analisis Pentingnya Standar Koreksi Geometrik Citra Satelit Resolusi Menengah Dan Kebutuhan Manfaat Bagi Pengguna. *Jurnal Standardisasi*, 21(1), 45–54. <https://doi.org/10.31153/js.v21i1.735>
- Macarof, P., & Statescu, F. (2017). Comparasion of NDBI and NDVI as Indicators of Surface Urban Heat Island Effect in Landsat 8 Imagery: A Case Study of Iasi. *Present Environment and Sustainable Development*, 11(2), 141–150. <https://doi.org/10.1515/pesd-2017-0032>
- Malik, M. S., Shukla, J. P., & Mishra, S. (2019). Relationship of LST, NDBI and NDVI Using Landsat-8 Data in Kandaihimmat Watershed, Hoshangabad, India. *Indian Journal of Geo-Marine Sciences*, 48(1), 25–31.
- Marsitha, F., Pattipeilohy, W. J., & Virgianto, R. H. (2019). Kenyamanan Termal Klimatologis Kota-Kota Besar Di Pulau Sulawesi Berdasarkan Temperature Humidity Index (Thi). *Jurnal Saintika UNPAM*, 1(2), 202–211. <https://doi.org/10.32493/jsmu.v1i2.2384>
- Maru, R., Baharuddin, I. I., Umar, R., Rasyid, R., Uca, Sanusi, W., & Bayudin. (2015). Analysis of The Heat Island Phenomenon in Makassar, South Sulawesi, Indonesia. *American Journal of Applied Sciences*. <https://doi.org/10.3844/ajassp.2015>
- Mirnayani, Rapang, S. K., Aini, A. N., & Bayanuddin, A. A. (2021). Pemanfaatan Data Citra Sentinel-3 Sea and Land Surface Temperature Radiometer (SLSTR) Pagi dan Malam Hari Untuk Analisis Intensitas Fenomena Pulau Bahang Permukaan (Studi Kasus : Kota Bandung). *Jurnal Penginderaan Jauh*, 18(1), 15–25.
- Moser, G., Serpico, S. B., & Benediktsson, J. A. (2013). Land-Cover Mapping by Markov Modeling of Spatial-Contextual Information in Very-High-Resolution Remote Sensing Images. *Proceedings of the IEEE*, 101(3), 631–651. <https://doi.org/10.1109/JPROC.2012.2211551>
- Mushore, T. D., Odindi, J., Dube, T., & Mutanga, O. (2017). Prediction of Future Urban Surface Temperatures Using Medium Resolution Satellite Data in Harare Metropolitan City, Zimbabwe. *Building and Environment*. <https://doi.org/10.1016/j.buildenv.2017.06.033>
- NASA. (2020a). *Landsat Homepage / Landsat Science*. <https://landsat.gsfc.nasa.gov/>

- NASA. (2020b). *What Is an Urban Heat Island? / NASA Climate Kids.* <https://climatekids.nasa.gov/heat-islands/>
- NOAA. (2018). *What is the Difference Between Land Cover and Land Use?* National Oceanic and Atmospheric Administration. <http://oceanservice.noaa.gov/facts/lclu.html>
- Noi Phan, T., Kuch, V., & Lehnert, L. W. (2020). Land Cover Classification Using Google Earth Engine and Random Forest Classifier-The Role of Image Composition. *Remote Sensing*, 12(2411), 1–22. <https://doi.org/10.3390/RS12152411>
- Paharuddin. (2012). Simulasi Geospasial Berbasis Cellular Automata Perubahan Penggunaan Lahan Untuk Prediksi Sedimentasi. In *Disertasi*. Universitas Hasanuddin.
- Parhusip, J. (2019). Penerapan Metode Analytical Hierarchy Process (AHP) Pada Desain Sistem Pendukung Keputusan Pemilihan Calon Penerima Bantuan Pangan Non Tunai (BPNT) Di Kota Palangka Raya. *Jurnal Teknologi Informasi*, 13(2), 18–29. <https://doi.org/10.47111/jti.v13i2.251>
- Peng, S., Piao, S., Ciais, P., Friedlingstein, P., Ottle, C., Bréon, F. M., Nan, H., Zhou, L., & Myneni, R. B. (2012). Surface urban heat island across 419 global big cities. *Environmental Science and Technology*. <https://doi.org/10.1021/es2030438>
- Peruge, T. V. D. (2013). Model Perubahan Penggunaan Lahan Menggunakan Cellular Automata-Markov Chain di Kawasan Mamminasata. In *Skripsi*. Universitas Hasanuddin.
- Pichierri, M., Bonafoni, S., & Biondi, R. (2012). Satellite air temperature estimation for monitoring the canopy layer heat island of Milan. *Remote Sensing of Environment*. <https://doi.org/10.1016/j.rse.2012.08.025>
- Purwadhi, S. H., & Sanjoto, T. B. (2008). *Pengantar Interpretasi Citra Penginderaan Jauh*. Pusat Data Penginderaan Jauh LAPAN dan Geografi Universitas Negeri Semarang.
- Rahmadewi, D. P. (2019). Kajian Perubahan Penutup Lahan Dengan Pemodelan Cellular Automata dan Pengaruhnya Terhadap Suhu Permukaan Lahan di Kabupaten Semarang. In *Skripsi*. Universitas Negeri Semarang.
- Rahmadewi, D. P., & Hanafi, F. (2020). Kajian Perubahan Penutup Lahan Dengan Pemodelan Cellular Automata dan Pengaruhnya Terhadap Suhu Permukaan Lahan di Kabupaten Semarang. *Geo Image*, 9(2), 154–166.
- Rahmah, A. N., Subiyanto, S., & Amarrohman, F. J. (2020). Pemodelan Perubahan Penggunaan Lahan Dengan Artificial Neural Network (ANN) di Kota Semarang. *Jurnal Geodesi Undip*, 9(1), 197–206.
- Rahman, M. T., Aldosary, A. S., & Mortoja, M. G. (2017). Modeling Future Land Cover Changes and Their Effects on the Land Surface Temperatures in the Saudi Arabian Eastern Coastal City of Dammam. *Land*, 6(36), 1–16. <https://doi.org/10.3390/land6020036>
- Ravanelli, R., Nascetti, A., Cirigliano, R. V., Di Rico, C., Leuzzi, G., Monti, P., & Crespi, M. (2018). Monitoring the impact of land cover change on surface urban heat island through Google Earth Engine: Proposal of a global methodology, first applications and problems. *Remote Sensing*, 10(9), 1–21.

- <https://doi.org/10.3390/rs10091488>
- Renard, F., Alonso, L., Fitts, Y., Hadjiosif, A., & Comby, J. (2019). Evaluation of the effect of urban redevelopment on surface urban heat islands. *Remote Sensing*, 11(3), 1–31. <https://doi.org/10.3390/rs11030299>
- Rocha, J., Ferreira, J. C., Simões, J., & Tenedório, J. A. (2007). Modelling Coastal and Land Use Evolution Patterns Through Neural Network and Cellular Automata Integration. *Journal of Coastal Research*, 827–831.
- Saaty, R. W. (1987). The Analytic Hierarchy Process-What It Is and How It Is Used. *Mathematical Modelling*, 9(3–5), 161–176. [https://doi.org/10.1016/0270-0255\(87\)90473-8](https://doi.org/10.1016/0270-0255(87)90473-8)
- Sambodo, K. A., Rahayu, M. I., Indriasari, N., & Natsir, M. (2014). Klasifikasi Hutan-Non Hutan Data Alos Palsar Menggunakan Metode Random Forest. *Prosiding Seminar Nasional Penginderaan Jauh 2014*, 120–127.
- Sejati, A. W., Buchori, I., & Rudiarto, I. (2019). The Spatio-Temporal Trends of Urban Growth and Surface Urban Heat Islands Over Two Decades in the Semarang Metropolitan Region. *Sustainable Cities and Society*. <https://doi.org/10.1016/j.scs.2019.101432>
- Sharifi, E., & Lehmann, S. (2015). Correlation Analysis of Surface Temperature of Rooftops, Streetscapes and Urban Heat Island Effect: Case Study of Central Sydney. *Journal of Urban and Environmental Engineering*, 9(1), 3–11. <https://doi.org/10.4090/juee.2015.v9n1.003011>
- Shastri, H., Barik, B., Ghosh, S., Venkataraman, C., & Sadavarte, P. (2017). Flip flop of Day-night and Summer-Winter Surface Urban Heat Island Intensity in India. *Scientific Reports*, 7(40178), 1–8. <https://doi.org/10.1038/srep40178>
- Shorabeh, S. N., Hamzeh, S., Shahraki, S. Z., Firozjaei, M. K., & Arsanjani, J. J. (2020). Modelling the Intensity of Surface Urban Heat Island and Predicting the Emerging Patterns: Landsat Multi-Temporal Images and Tehran As Case Study. *International Journal of Remote Sensing*, 41(19), 7384–7410. <https://doi.org/10.1080/01431161.2020.1759841>
- Shumilo, L., Kussul, N., Shelestov, A., Korsunska, Y., & Yailymov, B. (2019). Sentinel-3 Urban Heat Island Monitoring and analysis for Kyiv Based on Vector Data. *2019 10th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, 131–135. <https://doi.org/10.1109/DESSERT.2019.8770042>
- Sobirin, & Fatimah, R. N. (2015). Urban Heat Island Kota Surabaya. *Geoedukasi*, 4(2), 46–69.
- Sobrino, J. A., Oltra-Carrió, R., Sòria, G., Bianchi, R., & Paganini, M. (2012). Impact of spatial resolution and satellite overpass time on evaluation of the surface urban heat island effects. *Remote Sensing of Environment*, 117, 50–56. <https://doi.org/10.1016/j.rse.2011.04.042>
- Sobrino, José Antonio, & Irakulis, I. (2020). A methodology for comparing the surface urban heat Island in selected urban agglomerations around the world from Sentinel-3 SLSTR data. *Remote Sensing*, 12(12). <https://doi.org/10.3390/RS12122052>
- Song, Y., & Wu, C. (2016). Examining the impact of Urban Biophysical Composition and Neighboring Environment on Surface Urban Heat Island

- Effect. *Advances in Space Research*, 57, 96–109.
<https://doi.org/10.1016/j.asr.2015.10.036>
- Stathopoulou, M., & Cartalis, C. (2009). Downscaling AVHRR Land Surface Temperatures for Improved Surface Urban Heat Island Intensity Estimation. *Remote Sensing of Environment*, 2592–2605.
<https://doi.org/10.1016/j.rse.2009.07.017>
- Sugandi, D. (2010). *Bahan Pembelajaran Penginderaan Jauh*.
http://file.upi.edu/Direktori/FPIPS/JUR._PEND._GEOGRAFI/195805261986031-DEDE_SUGANDI/Sil_PJ.pdf
- Suriana, D., Barkey, R. A., & Gou, Z. (2020). Analysis of Land Use/Land Cover Change And Their Effects On Spatiotemporal Patterns Of Urban Heat Islands (UHI) In The City Of Makassar , Indonesia. *International Journal of Engineering and Science Applications*, 7(2), 113–123.
- Susilo, B. (2011). Pemodelan Spasial Probabilistik Integrasi Markov Chain Dan Cellular Automata Untuk Kajian Perubahan Penggunaan Lahan Skala Regional Di Provinsi Daerah Istimewa Yogyakarta. *Jurnal Geografi Gea*, 11(2), 163–178. <https://doi.org/10.17509/gea.v11i2.1638>
- Syamsuddin, I., & Hwang, J. (2009). The Application of AHP Model to Guide Decision Makers: A Case Study of E-Banking Security. *ICCIT 2009 - 4th International Conference on Computer Sciences and Convergence Information Technology*. <https://doi.org/10.1109/ICCIT.2009.251>
- Tan, J., Yu, D., Li, Q., Tan, X., & Zhou, W. (2020). Spatial Relationship Between Land-Use/Land-Cover Change and Land Surface Temperature in The Dongting Lake Area, China. *Scientific Reports*, 10(9245), 1–9.
<https://doi.org/10.1038/s41598-020-66168-6>
- Tariq, A., & Shu, H. (2020). CA-Markov Chain Analysis of Seasonal Land Surface Temperature and Land Use Land Cover Change Using Optical Multi-Temporal Satellite Data of Faisalabad, Pakistan. *Remote Sensing*, 12(3402), 1–22. <https://doi.org/10.3390/rs12203402>
- To'sambo, Y. (2018). Analisis Geospasial Prediksi Ketersediaan Lahan Permukiman Kota Makassar Menggunakan Metode Cellular Automata. In *Skripsi*. Universitas Hasanuddin.
- Ullah, S., Ahmad, K., Sajjad, R. U., Abbasi, A. M., Nazeer, A., & Tahir, A. A. (2019). Analysis and Simulation of Land Cover Changes and Their Impacts on Land Surface Temperature in A Lower Himalayan Region. *Journal of Environmental Management*, 245, 348–357.
<https://doi.org/10.1016/j.jenvman.2019.05.063>
- Ullah, S., Tahir, A. A., Akbar, T. A., Hassan, Q. K., Dewan, A., Khan, A. J., & Khan, M. (2019). Remote Sensing-Based Quantification of The Relationships Between Land Use Land Cover Changes and Surface Temperature Over The Lower Himalayan region. *Sustainability*, 11(5492), 1–16.
<https://doi.org/10.3390/su11195492>
- United Nations. (2015). *Transforming Our World : The 2030 Agenda for Sustainable Development*. <http://www.un.org/>
- United Nations. (2018). *2018 Revision of World Urbanization Prospects / Multimedia Library - United Nations Department of Economic and Social*

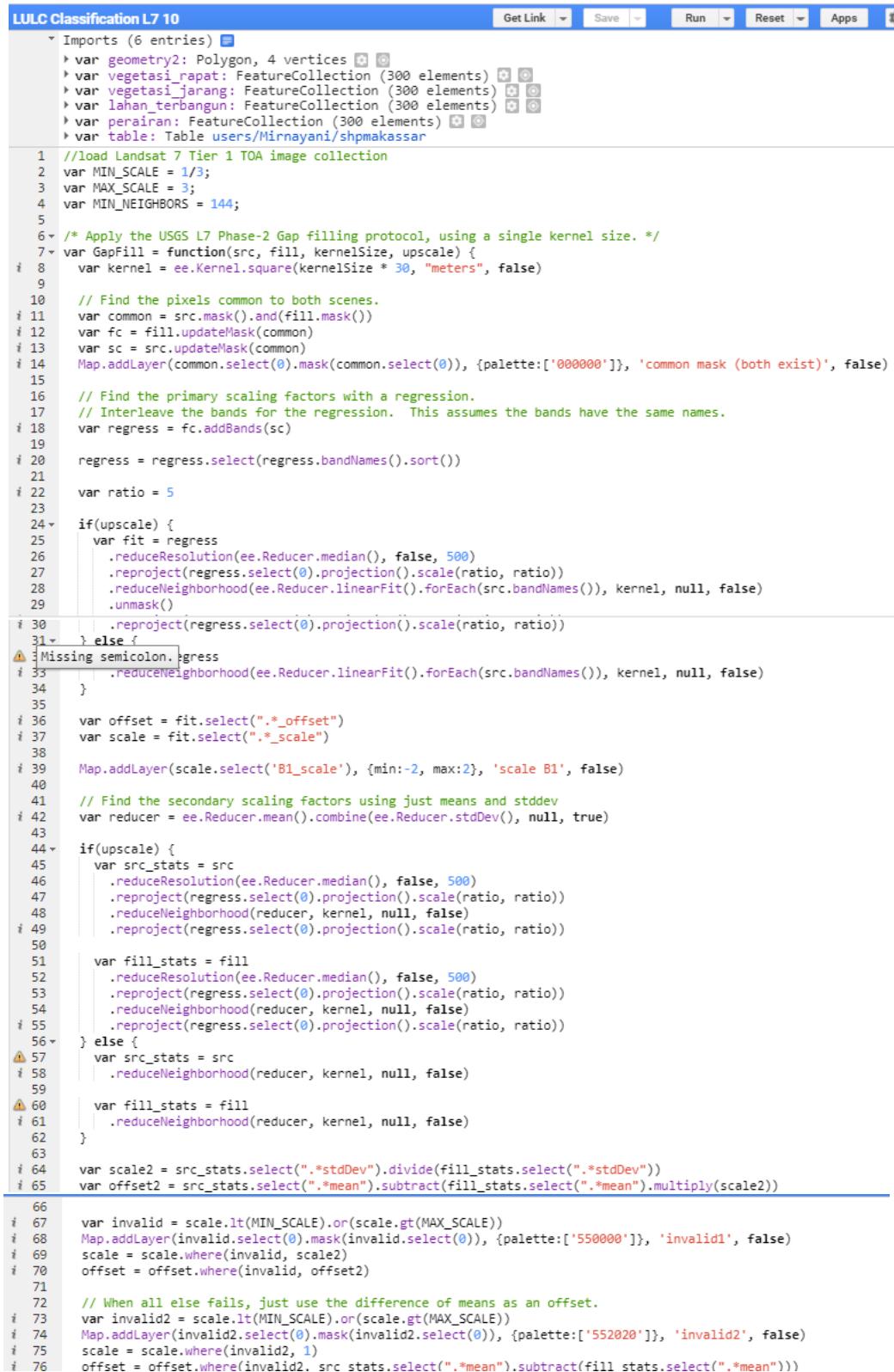
- Affairs. United Nations. <https://www.un.org/development/desa/publications/2018-revision-of-world-urbanization-prospects.html>
- USGS. (2016). *Landsat 8 Data Users Handbook*. <https://landsat.usgs.gov/documents/Landsat8DataUsersHandbook.pdf>
- Wang, K., Jiang, S., Wang, J., Zhou, C., Wang, X., & Lee, X. (2017). Comparing the diurnal and seasonal variabilities of atmospheric and surface urban heat islands based on the Beijing urban meteorological network. *Journal of Geophysical Research*, 122(4), 2131–2154. <https://doi.org/10.1002/2016JD025304>
- Wang, S. Q., Zheng, X. Q., & Zang, X. B. (2012). Accuracy Assessments of Land Use Change Simulation Based on Markov-Cellular Automata Model. *Procedia Environmental Sciences*, 13, 1238–1245. <https://doi.org/10.1016/j.proenv.2012.01.117>
- Wardana, I. K. (2015). *Analysis of Urban Surface Temperature For Green Spaces Planning In Bandung City, Indonesia*. University of Twente.
- Weng, Q. (2009). Thermal Infrared Remote Sensing for Urban Climate and Environmental Studies: Methods, applications, and Trends. *ISPRS Journal of Photogrammetry and Remote Sensing*, 64, 335–344. <https://doi.org/10.1016/j.isprsjprs.2009.03.007>
- Weng, Q., Firozjaei, M. K., Sedighi, A., Kiavarz, M., & Alavipanah, S. K. (2018). Statistical Analysis of Surface Urban Heat Island Intensity Variations: A Case Study of Babol city, Iran. *GIScience and Remote Sensing*, 1–29. <https://doi.org/10.1080/15481603.2018.1548080>
- WHO. (2012). WHO | Urbanization and health. In *Who*. <http://www.who.int/globalchange/ecosystems/urbanization/en/>
- Widiastusi, J. (2018). Klasifikasi Pembiayaan Warung Mikro Menggunakan Metode Random Forest dengan Teknik Sampling Kelas Imbalanced. In *Skripsi*. Universitas Islam Indonesia.
- Wiweka. (2014). Pola Suhu Permukaan dan Udara Menggunakan Citra Satelit Landsat Multitemporal. *Jurnal Ecolab*, 8(1), 11–22. <https://doi.org/10.20886/jklh.2014.8.1.11-22>
- Wulder, M. A., White, J. C., Loveland, T. R., Woodcock, C. E., Belward, A. S., Cohen, W. B., Fosnight, E. A., Shaw, J., Masek, J. G., & Roy, D. P. (2015). The Global Landsat Archive: Status, Consolidation, and Direction. *Remote Sensing of Environment*, 1–13. <https://doi.org/10.1016/j.rse.2015.11.032>
- Yao, N., Huang, C., Yang, J., Bosch, C. C. K. van den, Ma, L., & Jia, Z. (2020). Combined Effects of Impervious Surface Change and Large-scale Afforestation on the Surface Urban Heat Island Intensity of Beijing, China Based on Remote Sensing Analysis. *Remote Sensing*, 12(3906), 1–22. <https://doi.org/10.3390/rs12233906>
- Yu, Z., Guo, X., Zeng, Y., Koga, M., & Vejre, H. (2018). Variations in Land Surface Temperature and Cooling Efficiency of Green Space in Rapid Urbanization: The Case of Fuzhou city, China. *Urban Forestry and Urban Greening*, 29, 113–121. <https://doi.org/10.1016/j.ufug.2017.11.008>
- Yuan, Y., Xi, C., Jing, Q., & Felix, N. (2017). Seasonal Variations of the Urban

- Thermal Environment Effect in a Tropical Coastal City. *Advances in Meteorology*, 2017. <https://doi.org/10.1155/2017/8917310>
- Yudarwati, R., Sitorus, S. R. ., & Munibah, K. (2016). Arahan Pengendalian Perubahan Penggunaan Lahan Menggunakan Markov - Cellular Automata Di Kabupaten Cianjur. *Tataloka*, 18(4), 211–221. <https://doi.org/10.14710/tataloka.18.4.211-221>
- Zhou, B., Rybski, D., & Kropf, J. P. (2017). The Role of City Size and Urban Form in The Surface Urban Heat Island. *Scientific Reports*, 7(4791), 1–9. <https://doi.org/10.1038/s41598-017-04242-2>
- Zhou, D., Xiao, J., Bonafoni, S., Berger, C., Deilami, K., Zhou, Y., Frolking, S., Yao, R., Qiao, Z., & Sobrino, J. A. (2019). Satellite remote sensing of surface urban heat islands: Progress, challenges, and perspectives. *Remote Sensing*, 11(1), 1–36. <https://doi.org/10.3390/rs11010048>

LAMPIRAN

LAMPIRAN 1 : Pengolahan Data Tutupan Lahan

1. Script Klasifikasi Tutupan Lahan Tahun 2010



The screenshot shows a Jupyter Notebook interface with the title "LULC Classification L7 10". The code cell contains a complex Earth Engine script for land cover classification. The script includes imports, variable definitions, and various processing steps involving reducers, projections, and neighborhood analysis. There are several warning messages in the code, such as "Missing semicolon" and "invalid syntax", indicating potential errors or incomplete code.

```
LULC Classification L7 10
Get Link Save Run Reset Apps

Imports (6 entries)
var geometry2: Polygon, 4 vertices
var vegetasi_rapat: FeatureCollection (300 elements)
var vegetasi_jarang: FeatureCollection (300 elements)
var lahan_terbangun: FeatureCollection (300 elements)
var perairan: FeatureCollection (300 elements)
var table: Table users/Mirnayani/shpmakassar

1 //Load Landsat 7 Tier 1 TOA image collection
2 var MIN_SCALE = 1/3;
3 var MAX_SCALE = 3;
4 var MIN_NEIGHBORS = 144;
5
6 /* Apply the USGS L7 Phase-2 Gap filling protocol, using a single kernel size. */
7 var GapFill = function(src, fill, kernelSize, upscale) {
8     var kernel = ee.Kernel.square(kernelSize * 30, "meters", false)
9
10    // Find the pixels common to both scenes.
11    var common = src.mask().and(fill.mask())
12    var fc = fill.updateMask(common)
13    var sc = src.updateMask(common)
14    Map.addLayer(common.select(0).mask(common.select(0)), {palette:['000000']}, 'common mask (both exist)', false)
15
16    // Find the primary scaling factors with a regression.
17    // Interleave the bands for the regression. This assumes the bands have the same names.
18    var regress = fc.addBands(sc)
19
20    regress = regress.select(regress.bandNames().sort())
21
22    var ratio = 5
23
24    if(upscale) {
25        var fit = regress
26            .reduceResolution(ee.Reducer.median(), false, 500)
27            .reproject(regress.select(0).projection().scale(ratio, ratio))
28            .reduceNeighborhood(ee.Reducer.linearFit().forEach(src.bandNames()), kernel, null, false)
29            .unmask()
30            .reproject(regress.select(0).projection().scale(ratio, ratio))
31    } else {
32        var offset = fit.select(".*_offset")
33        var scale = fit.select(".*_scale")
34    }
35
36    var offset2 = src.select(".*_offset")
37    var scale2 = src.select(".*_scale")
38
39    Map.addLayer(scale.select('B1_scale'), {min:-2, max:2}, 'scale B1', false)
40
41    // Find the secondary scaling factors using just means and stddev
42    var reducer = ee.Reducer.mean().combine(ee.Reducer.stdDev(), null, true)
43
44    if(upscale) {
45        var src_stats = src
46            .reduceResolution(ee.Reducer.median(), false, 500)
47            .reproject(regress.select(0).projection().scale(ratio, ratio))
48            .reduceNeighborhood(reducer, kernel, null, false)
49            .reproject(regress.select(0).projection().scale(ratio, ratio))
50
51        var fill_stats = fill
52            .reduceResolution(ee.Reducer.median(), false, 500)
53            .reproject(regress.select(0).projection().scale(ratio, ratio))
54            .reduceNeighborhood(reducer, kernel, null, false)
55            .reproject(regress.select(0).projection().scale(ratio, ratio))
56    } else {
57        var src_stats = src
58            .reduceNeighborhood(reducer, kernel, null, false)
59
60        var fill_stats = fill
61            .reduceNeighborhood(reducer, kernel, null, false)
62    }
63
64    var scale2 = src_stats.select(".*stdDev").divide(fill_stats.select(".*stdDev"))
65    var offset2 = src_stats.select(".*mean").subtract(fill_stats.select(".*mean")).multiply(scale2)
66
67    var invalid = scale.lt(MIN_SCALE).or(scale.gt(MAX_SCALE))
68    Map.addLayer(invalid.select(0).mask(invalid.select(0)), {palette:['550000']}, 'invalid1', false)
69    scale = scale.where(invalid, scale2)
70    offset = offset.where(invalid, offset2)
71
72    // When all else fails, just use the difference of means as an offset.
73    var invalid2 = scale.lt(MIN_SCALE).or(scale.gt(MAX_SCALE))
74    Map.addLayer(invalid2.select(0).mask(invalid2.select(0)), {palette:['552020']}, 'invalid2', false)
75    scale = scale.where(invalid2, 1)
76    offset = offset.where(invalid2, src.stats.select(".*mean").subtract(fill.stats.select(".*mean")))
```

```

77 // Apply the scaling and mask off pixels that didn't have enough neighbors.
78 var count = common.reduceNeighborhood(ee.Reducer.count(), kernel, null, true, "boxcar")
79 var scaled = fill.multiply(scale).add(offset)
80 || .updateMask(count.gte(MIN_NEIGHBORS))
81
82 return src.unmask(scaled, true)
83 }
84
85
86 var source = ee.Image("LANDSAT/LE07/C01/T1_TOA/LE07_114064_20100902")
87 var fill = ee.Image("LANDSAT/LE07/C01/T1_TOA/LE07_114064_20100310")
88
89 /*
90 Map.addLayer(source, {bands: ['B3', 'B2', 'B1'], min: 0, max: 0.3}, 'Makassar_2020src')
91 Map.addLayer(fill, {bands: ['B3', 'B2', 'B1'], min: 0, max: 0.3}, 'Makassar_2020fill');
92 */
93
94 /*
95 Map.addLayer(fill, {min:0, max:0.3, bands:["B3", "B2", "B1"]}, "fill", true)
96 Map.addLayer(source, {min:0, max:0.3, bands:["B3", "B2", "B1"]}, "destination", true)
97 */
98
99 var landsat7_2010 = GapFill(source, fill, 10, false);
100 Map.addLayer(landsat7_2010, {min:0, max:0.3, bands:["B3", "B2", "B1"]}, "filled", true),
101
102 var landsat7_2010 = GapFill(source, fill, 10, true);
103 Map.addLayer(landsat7_2010, {min:0, max:0.3, bands:["B3", "B2", "B1"]}, "filled (upscaled)", true)
104
105 /*
106 Map.addLayer(landsat7_2010, {bands: ['B7', 'B5', 'B3'], min: 0, max: 0.3}, 'Urban', true);
107 Map.addLayer(landsat7_2010, {bands: ['B4', 'B3', 'B2'], min: 0, max: 0.3}, 'Vegetation', true);
108 */
109
110
111 //Merge sample points together into one FeatureCollection
112 var landcover_2010 = vegetasi_rapat.merge(vegetasi_jarang).merge(lahan_terbangun).merge(perairan);
113
114 //Select Bands from mosaic Image for training
115 var bands = ['B1', 'B2', 'B3', 'B4', 'B5', 'B6_VCID_1', 'B6_VCID_2', 'B7'];
116
117 //The name of the property on the points storing the class lebel
118 var classProperty = 'landcover';
119
120 //Sample the input imagery to get a FeatureCollection of training data
121 var training = landsat7_2010.select(bands).sampleRegions({
122   collection: landcover_2010,
123   properties: [classProperty],
124   scale: 30
125 });
126
127 //Train the classifier
128 var classifier = ee.Classifier.smileRandomForest(10).train({
129   features: training,
130   classProperty: classProperty,
131 });
132
133 //Classify the input imagery
134 var classified_2010 = landsat7_2010.classify(classifier);
135
136 //Define color palette
137 var palette = [
138   '04831F', // vegetasi_rapat (0) // green {htmlcolorcodes.com}
139   '23F506', // vegetasi_jarang (1)
140   'F30707', //lahan terbangun (2) // red
141   '0724ed' //perairan (3) // blue
142 ];
143
144 //Display the classified result
145 Map.addLayer(classified_2010, {
146   min:0, max: 3, palette: palette}, 'LULC_2010');
147
148 // Optionally, do some accuracy assessment. First, add a column of random uniforms to the training dataset.
149 var withRandom = training.randomColumn('random');
150
151 // We want to reserve some of the data for testing, to avoid overfitting the model.
152 var split = 0.7; //Roughly 70% training, 30% testing.
153 var trainingPartition = withRandom.filter(ee.Filter.lt('random', split));
154 var testingPartition = withRandom.filter(ee.Filter.gte('random', split));
155
156 // Get a confusion matrix representing resubstitution accuracy.
157 var trainAccuracy = classifier.confusionMatrix();
158 print('Resubstitution error matrix: ', trainAccuracy);
159 print('Training overall accuracy: ', trainAccuracy.accuracy());
160
161 //classify the test FeatureCollection.
162 var test = testingPartition.classify(classifier);
163
164 // Get a confusion matrix representing expected accuracy.
165 var testAccuracy = test.errorMatrix(classProperty, 'classification');
166 print('Validation error matrix: ', testAccuracy);
167 print('Validation overall accuracy: ', testAccuracy.accuracy());

```

```

168 /*
169 * 
170 * Export.image.toDrive({
171   image: classified_2010,
172   description: 'NewData_Tutupanlahan_2010',
173   scale: 30,
174   region: table,
175   fileFormat: 'GeoTIFF',
176 });
177 */
178

```

2. Script Klasifikasi Tutupan Lahan tahun 2015

LULC Classification L8 15

```

1 //Load Landsat 8 Tier 1 TOA image collection
2 var landsat_2020 = ee.ImageCollection("LANDSAT/LC08/C01/T1_TOA")
3   .filterDate("2015-07-22", "2015-07-31")
4   .filter(ee.Filter.eq('WRS_PATH', 114))
5   .filter(ee.Filter.eq('WRS_ROW', 64))
6   .filter(ee.Filter.lt('CLOUD_COVER',10))
7   .sort('CLOUD_COVER')
8   .first()
9   .clip(geometry2);
10
11 print(landsat_2020);
12
13
14 Map.addLayer(landsat_2020, {bands: ['B4', 'B3', 'B2'], min: 0, max: 0.3}, 'Citra_2020');
15
16 //Visualisasi citra
17 Map.addLayer(landsat_2020, {bands: ['B7', 'B6', 'B4'], min: 0, max: 0.3}, 'Urban');
18 Map.addLayer(landsat_2020, {bands: ['B5', 'B4', 'B3'], min: 0, max: 0.3}, 'Vegetation');
19 Map.addLayer(landsat_2020, {bands: ['B6', 'B5', 'B2'], min: 0, max: 0.3}, 'Agriculture');
20 Map.addLayer(landsat_2020, {bands: ['B5', 'B6', 'B4'], min: 0, max: 0.3}, 'Water');
21
22
23 //Merge sample points together into one FeatureCollection
24 var landcover_2020 = vegetasi_rapat.merge(vegetasi_jarang).merge(lahan_terbangun).merge(perairan);
25
26 //Select Bands from mosaic Image for training
27 var bands = ['B2', 'B3', "B4", 'B5', 'B7'];
28
29 //The name of the property on the points storing the class label
30 var classProperty = 'landcover';
31
32 //Sample the input imagery to get a FeatureCollection of training data
33 var training_2020 = landsat_2020.select(bands).sampleRegions({
34   collection: landcover_2020,
35   properties: [classProperty],
36   scale: 30
37 });
38
39 //Train the classifier
40 var classifier = ee.Classifier.smileRandomForest(10).train({
41   features: training_2020,
42   classProperty: classProperty,
43 });
44
45 //Classify the input imagery
46 var classified_2020 = landsat_2020.classify(classifier);
47
48 //Define color palette
49 var palette = [
50   '04831F', // vegetasi_rapat (0) // green {htmlcolorcodes.com}
51   '23F506', // vegetasi_jarang (1)
52   'F30707', // lahan_terbangun (2) // red
53   '0724ed' // perairan (3) // blue
54 ];
55
56 //Display the classified result
57 Map.addLayer(classified_2020, {
58   min:0, max: 3, palette: palette}, 'LULC_2020');
59
60
61 //Optionally, do some accuracy assessment. First, add a column of random uniforms to the training dataset.
62 var withRandom = training_2020.randomColumn('random');
63
64 //We want to reserve some of the data for testing, to avoid overfitting the model.
65 var split = 0.7; //Roughly 70% training, 30% testing.
66 var trainingPartition = withRandom.filter(ee.Filter.lt('random', split));
67 var testingPartition = withRandom.filter(ee.Filter.gte('random', split));
68

```

```

69 // Get a confusion matrix representing resubstitution accuracy.
70 var trainAccuracy_2020 = classifier.confusionMatrix();
71 print('Resubstitution error matrix_2020: ', trainAccuracy_2020);
72 print('Training overall accuracy_2020: ', trainAccuracy_2020.accuracy());
73
74 //classify the test FeatureCollection.
75 var test_1 = testingPartition.classify(classifier);
76
77 // Get a confusion matrix representing expected accuracy.
78 var testAccuracy_2020 = test_1.errorMatrix(classProperty, 'classification');
79 print('Validation error matrix_2020: ', testAccuracy_2020);
80 print('Validation overall accuracy_2020: ', testAccuracy_2020.accuracy());
81
82 /*
83 //Determine the Land Cover Change between 2001 and 2019
84 var LULC_Change = classified_2020.subtract(classified_2015);
85 Map.addLayer(LULC_Change.clip(table), {
86   min:0, max:2, palette: ['green', 'red', 'blue']}, 'LULC Change Between 2015 and 2020_Makassar');
87 */
88
89
90 /*
91 Export.image.toDrive({
92   image: classified_2020,
93   description: 'NewData_Tutupanlahan_2015',
94   scale: 30,
95   region: table,
96   fileFormat: 'GeoTIFF',
97 });
98 */

```

3. Script Klasifikasi Tutupan Lahan tahun 2020

LULC Classification LB 20

```

Imports (6 entries) ↗
↳ var geometry2: Polygon, 4 vertices ↗ ↗
↳ var vegetasi_rapat: FeatureCollection (300 elements) ↗ ↗
↳ var vegetasi_jarang: FeatureCollection (300 elements) ↗ ↗
↳ var lahan_terbangun: FeatureCollection (300 elements) ↗ ↗
↳ var perairan: FeatureCollection (300 elements) ↗ ↗
↳ var table: Table users/Mirnayani/shpmakassar

1 //Load Landsat 8 Tier 1 TOA image collection
2 var landsat_2020 = ee.ImageCollection("LANDSAT/LC08/C01/T1_TOA")
3   .filterDate("2020-08-01", "2020-12-31")
4   .filter(ee.Filter.eq('WRS_PATH', 114))
5   .filter(ee.Filter.eq('WRS_ROW', 64))
6   .filter(ee.Filter.lt('CLOUD_COVER',10))
7   .sort('CLOUD_COVER')
8   .first()
9   .clip(geometry2);
10
11 print(landsat_2020);
12
13
14 Map.addLayer(landsat_2020, {bands: ['B4', 'B3', 'B2'], min: 0, max: 0.3}, 'Citra_2020');
15
16 //Visualisasi citra
17 Map.addLayer(landsat_2020, {bands: ['B7', 'B6', 'B4'], min: 0, max: 0.3}, 'Urban');
18 Map.addLayer(landsat_2020, {bands: ['B5', 'B4', 'B3'], min: 0, max: 0.3}, 'Vegetation');
19 Map.addLayer(landsat_2020, {bands: ['B6', 'B5', 'B2'], min: 0, max: 0.3}, 'Agriculture');
20 Map.addLayer(landsat_2020, {bands: ['B5', 'B6', 'B4'], min: 0, max: 0.3}, 'Water');
21
22
23 //Merge sample points together into one FeatureCollection
24 var landcover_2020 = vegetasi_rapat.merge(vegetasi_jarang).merge(lahan_terbangun).merge(perairan);
25
26 //Select Bands from mosaic Image for training
27 var bands = ['B2', 'B3', 'B4', 'B5', 'B7'];
28
29 //The name of the property on the points storing the class label
30 var classProperty = 'landcover';
31
32 //Sample the input imagery to get a FeatureCollection of training data
33 var training_2020 = landsat_2020.select(bands).sampleRegions({
34   collection: landcover_2020,
35   properties: [classProperty],
36   scale: 30
37 });
38
39 //Train the classifier
40 var classifier = ee.Classifier.smileRandomForest(10).train({
41   features: training_2020,
42   classProperty: classProperty,
43 });
44
45 //Classify the input imagery
46 var classified_2020 = landsat_2020.classify(classifier);
47

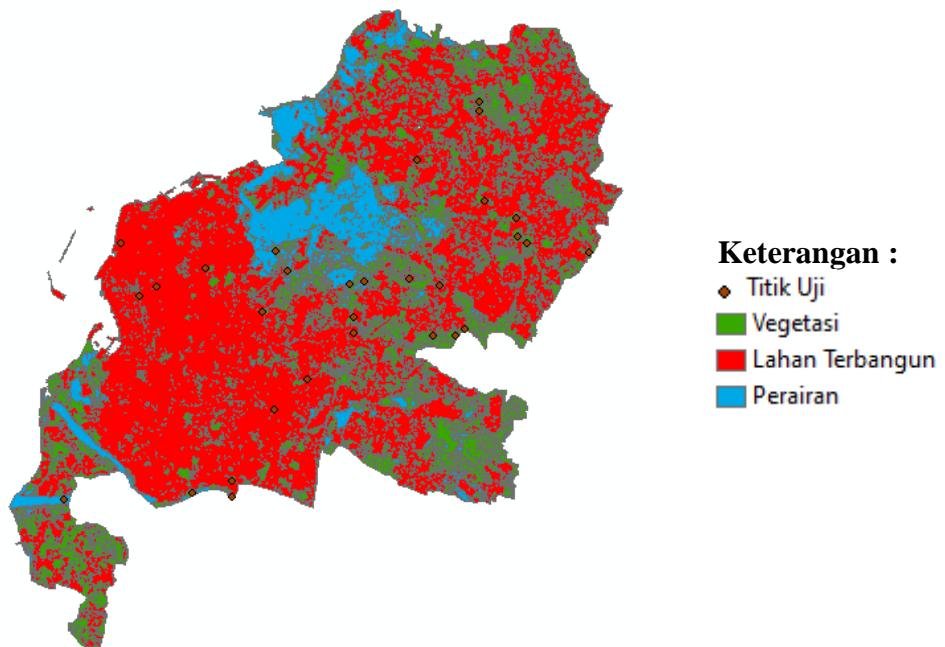
```

```

48 //Define color palette
49 var palette = [
50   '04831F', // vegetasi_rapat (0) // green {htmlcolorcodes.com}
51   '23F506', // vegetasi_jarang (1)
52   'F30707', // lahan_terbangun (2) // red
53   '0724ed' //perairan (3) // blue
54 ];
55
56 //Display the classified result
57 Map.addlayer(classified_2020, {
58   min:0, max: 3, palette: palette}, 'LULC_2020');
59
60
61 // Optionally, do some accuracy assessment. First, add a column of random uniforms to the training dataset.
62 var withRandom = training_2020.randomColumn('random');
63
64 // We want to reserve some of the data for testing, to avoid overfitting the model.
65 var split = 0.7; //Roughly 70% training, 30% testing.
66 var trainingPartition = withRandom.filter(ee.Filter.lt('random', split));
67 var testingPartition = withRandom.filter(ee.Filter.gte('random', split));
68
69 // Get a confusion matrix representing resubstitution accuracy.
70 var trainAccuracy_2020 = classifier.confusionMatrix();
71 print('Resubstitution error matrix_2020: ', trainAccuracy_2020);
72 print('Training overall accuracy_2020: ', trainAccuracy_2020.accuracy());
73
74 //classify the test FeatureCollection.
75 var test_1 = testingPartition.classify(classifier);
76
77 // Get a confusion matrix representing expected accuracy.
78 var testAccuracy_2020 = test_1.errorMatrix(classProperty, 'classification');
79 print('Validation error matrix_2020: ', testAccuracy_2020);
80 print('Validation overall accuracy_2020: ', testAccuracy_2020.accuracy());
81
82 /*
83 //Determine the Land Cover Change between 2001 and 2019
84 var LULC_Change = classified_2020.subtract(classified_2015);
85 Map.addlayer(LULC_Change.clip(table), {
86   min:0, max:2, palette: ['green', 'red', 'blue']], 'LULC Change Between 2015 and 2020_Makassar');
87 */
88
89
90 /*
91 Export.image.toDrive({
92   image: classified_2020,
93   description: 'NewData_Tutupanlahan_2020',
94   scale: 30,
95   region: table,
96   fileFormat: 'GeoTIFF',
97 });
98 */

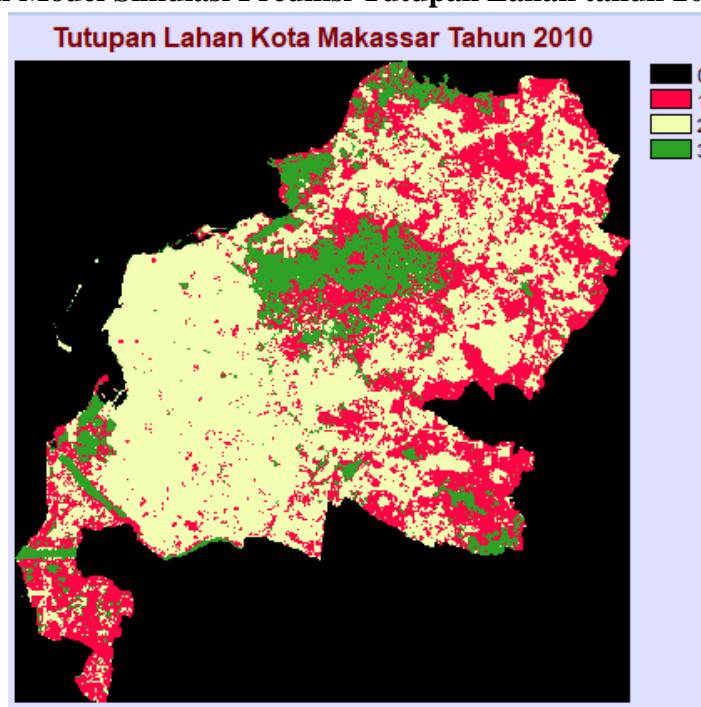
```

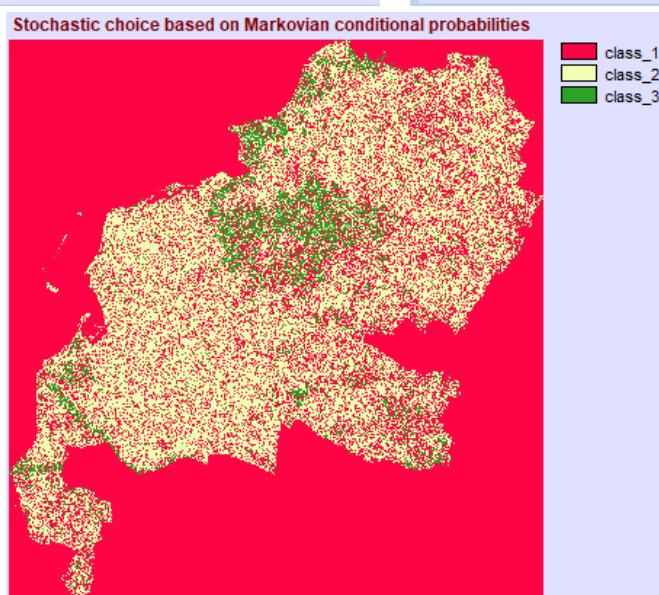
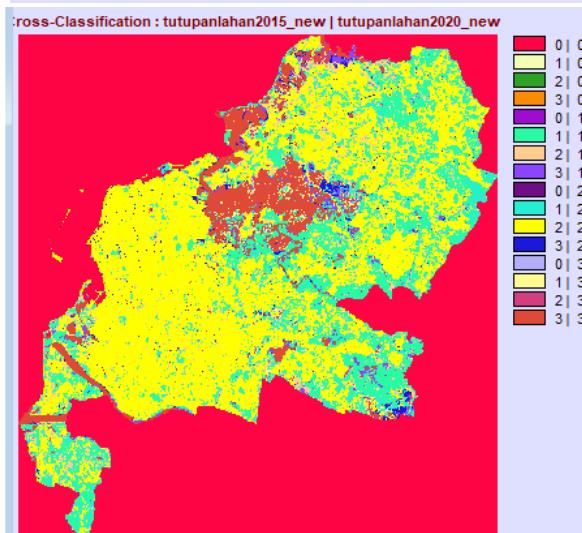
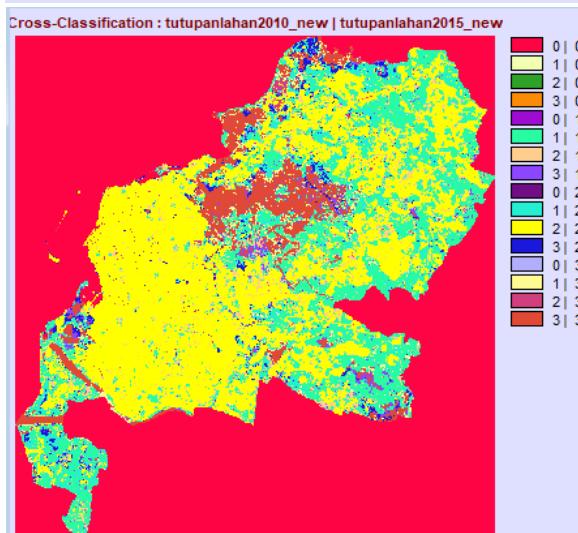
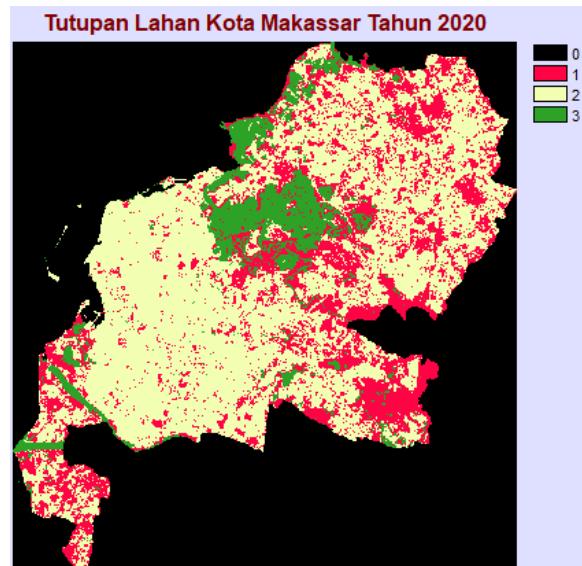
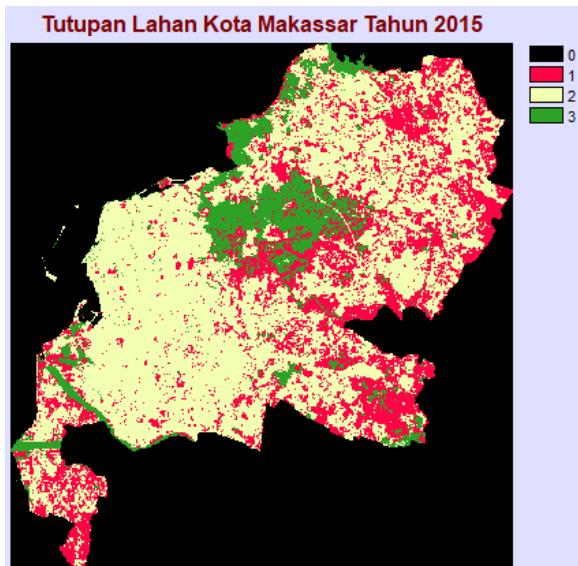
4. Titik Sampel Uji Akurasi Tutupan Lahan tahun 2010, 2015, 2020



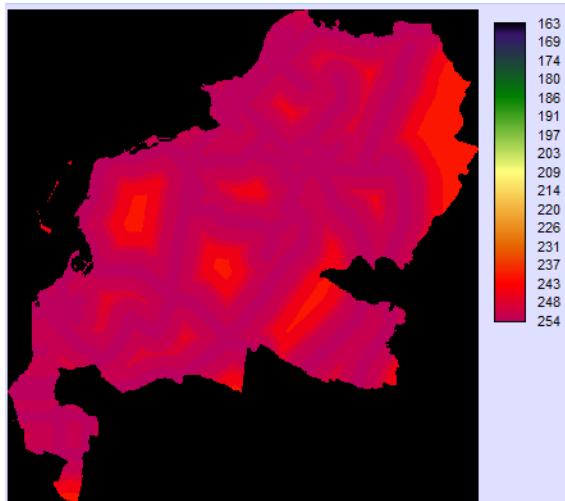
Titik Uji						
FID	Shape *	NAMOBJ	Validasi	User10	User15	User20
► 20	Point ZM	Vegetasi	1	1	1	1
21	Point ZM	Vegetasi	1	1	1	1
22	Point ZM	Sawah	1	1	1	1
23	Point ZM	Sawah	1	1	1	1
24	Point ZM	Sawah	1	3	1	1
25	Point ZM	Sawah	1	1	1	1
26	Point ZM	Sawah	1	1	2	1
27	Point ZM	Sawah	1	1	1	1
28	Point ZM	Sawah	1	1	1	1
29	Point ZM	Vegetasi	1	2	1	1
0	Point ZM	Monumen Maha Putra Emi Saelan	2	2	2	2
1	Point ZM	Monumen Mandala	2	2	2	2
2	Point ZM	Monumen 40000	2	2	2	2
10	Point ZM	PLTU Tello	2	2	2	2
14	Point ZM	RS Wahidin	2	2	2	2
15	Point ZM	Pelabuhan Soekarno Hatta	2	2	2	2
16	Point ZM	Terminal Daya	2	2	2	2
17	Point ZM	Terminal Malengkeri	2	2	2	2
18	Point ZM	Universitas Hasanuddin	2	2	2	1
19	Point ZM	SMA 1 Makassar	2	2	2	2
3	Point ZM	Sungai Jeneberang	3	3	3	3
4	Point ZM	Sungai Jeneberang	3	3	3	3
5	Point ZM	Sungai Jeneberang	3	3	3	3
6	Point ZM	Sungai Pampang	3	3	1	3
7	Point ZM	Sungai Pampang	3	3	3	3
8	Point ZM	Sungai Pampang	3	3	2	2
9	Point ZM	Sungai Pampang	3	2	3	2
11	Point ZM	Sungai Tello	3	1	3	3
12	Point ZM	Sungai Tello	3	1	3	3
13	Point ZM	Sungai Tello	3	3	3	3

5. Pengolahan Model Simulasi Prediksi Tutupan Lahan tahun 2025

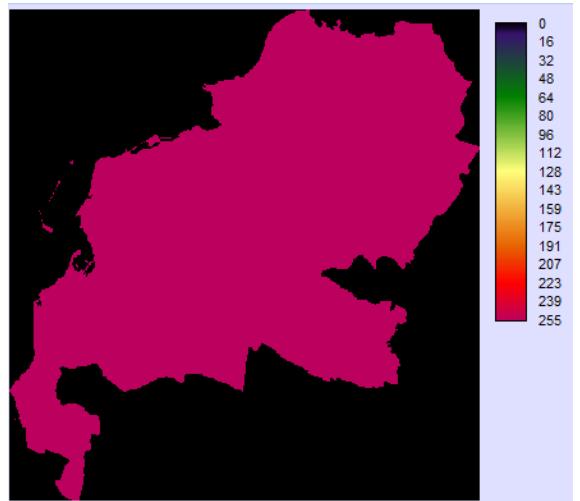




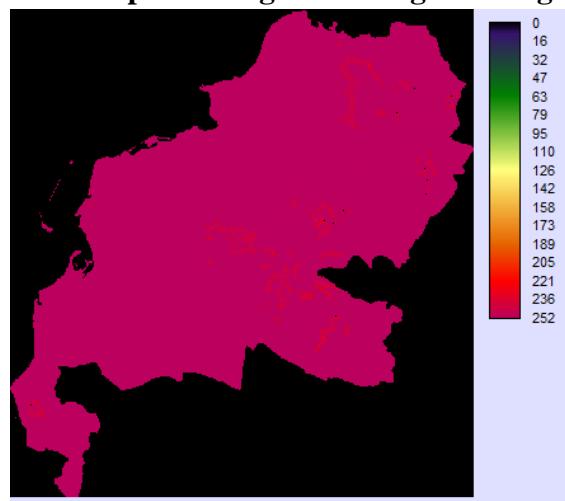
Faktor Pendorong : Jarak ke Jalan



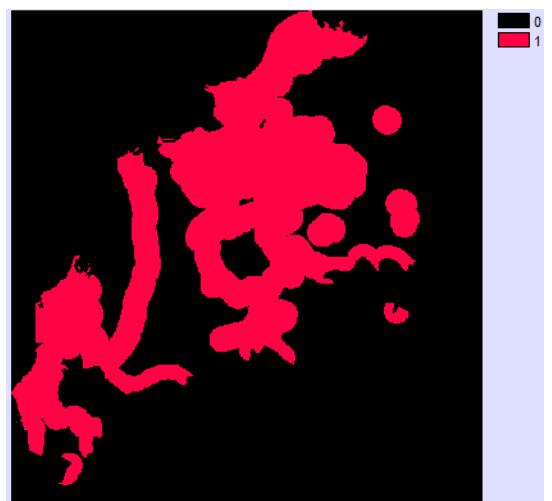
Faktor Pendorong : Jarak ke permukiman



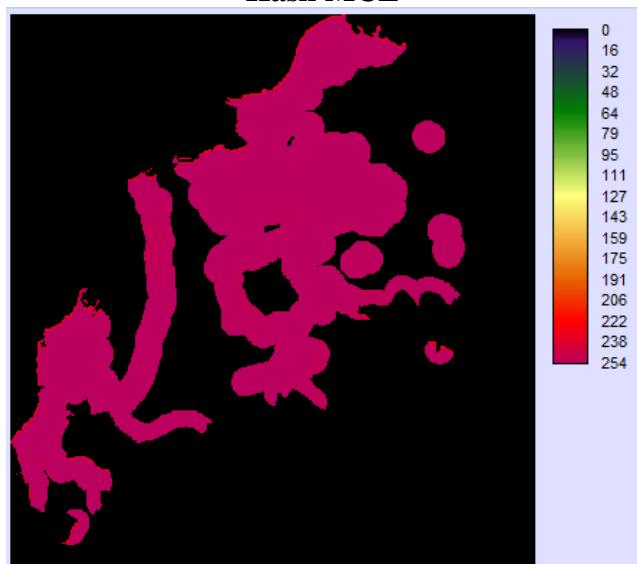
Faktor pendorong : kemiringan lereng



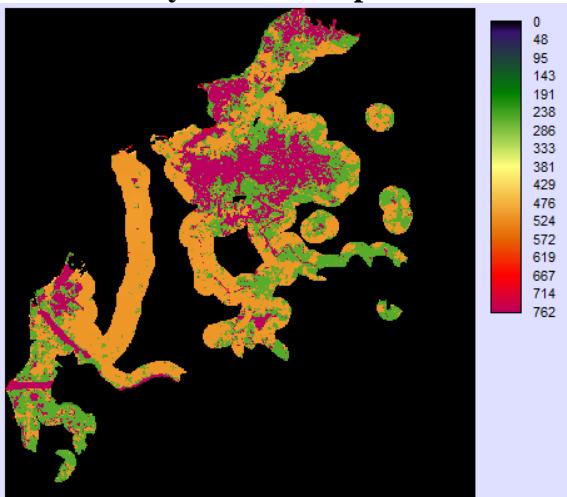
Faktor Pembatas : Tubuh air



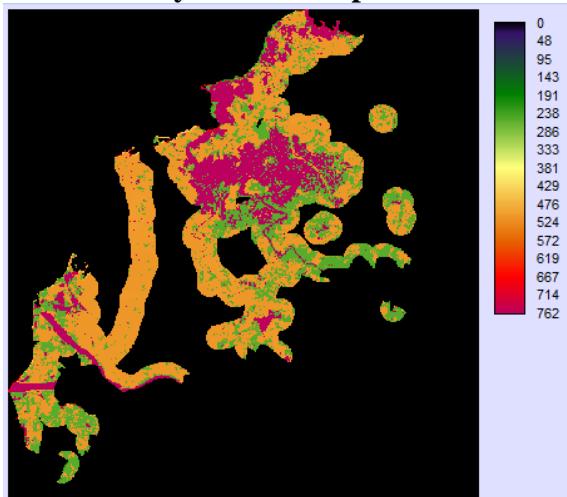
Hasil MCE



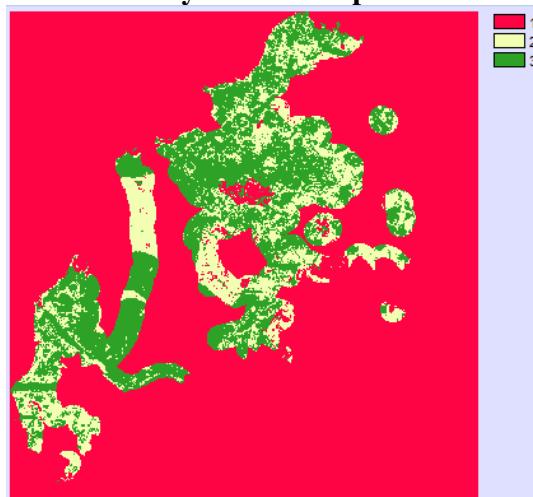
Hasil Overlay MCE- Tutuhan Lahan 2010



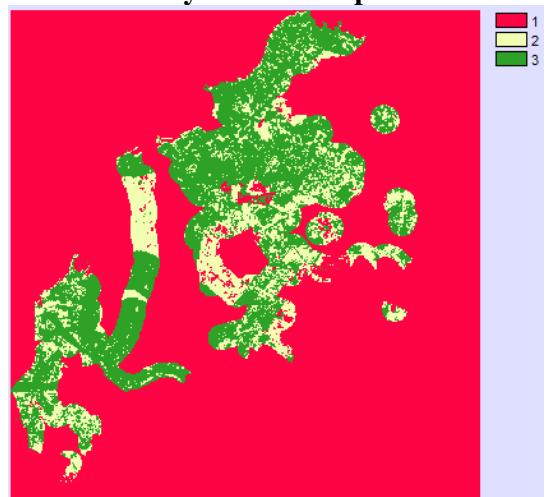
Hasil Overlay MCE-Tutuhan Lahan 2015



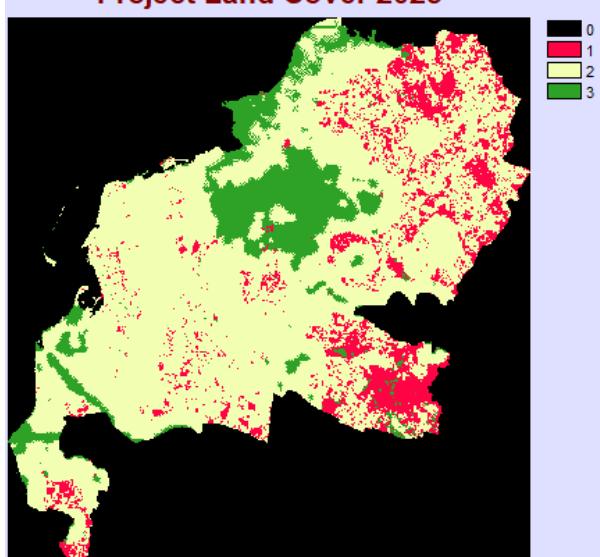
Reclass Overlay MCE-Tutuhan Lahan 2010



Reclass Overlay MCE-Tutuhan Lahan 2015



Project Land Cover 2025



LAMPIRAN 2 : Pengolahan Data LST

1. Script penentuan LST Tahun 2010

The screenshot shows the Google Earth Engine code editor interface. The title bar says "LST L7 single". The code itself is as follows:

```
LST L7 single
Get Link Save Run Reset Apps

Imports (1 entry)
var geometry: Polygon, 4 vertices

1 // link to the code that computes the Landsat LST
2 var LandsatLST = require('users/Mirnayani/GEE:Skripsi/Landsat_LST.js')
3
4
5 // select region of interest, date range, and landsat satellite
6 var satellite = 'L7';
7 var date_start = '2010-09-02';
8 var date_end = '2010-09-03';
9 var use_ndvi = true;
10
11 // get landsat collection with added variables: NDVI, FVC, TPW, EM, LST
12 var LandsatColl = LandsatLST.collection(satellite, date_start, date_end, geometry, use_ndvi)
13   .filter(ee.Filter.eq('WRS_PATH', 114))
14   .filter(ee.Filter.eq('WRS_ROW', 64))
15   .filter(ee.Filter.lt('CLOUD_COVER',10))
16
17 print(LandsatColl)
18
19 // select the first feature
20 var exImage = LandsatColl.first();
21 print(exImage)
22
23 var fill_exImage= exImage.focal_mean(1, "square", "pixels", 20)
24 var final_image = fill_exImage.blend(exImage)
25
26 var cmap1 = ['blue', 'green', 'yellow', 'orange', 'red'];
27 var cmap2 = ['F2F2F2','EFC2B3','ECB176','E9BD3A','E6E600','63C600','00A600'];
28
29 Map.centerObject(geometry, 10)
30 Map.addLayer(final_image.select('TPW'),{min:0.0, max:60.0, palette:cmap1},'TCW')
31 Map.addLayer(final_image.select('TPWpos'),{min:0.0, max:9.0, palette:cmap1},'TCWpos')
32 Map.addLayer(final_image.select('FVC'),{min:0.0, max:1.0, palette:cmap2},'FVC')
33 Map.addLayer(final_image.select('EM'),{min:0.9, max:1.0, palette:cmap1}, 'Emissivity')
34 Map.addLayer(final_image.select('B6_VCID_2'),{min:290, max:320, palette:cmap1}, 'TIR BT')
35 Map.addLayer(final_image.select('LST'),{min:290, max:320, palette:cmap1}, 'LST')
36 Map.addLayer(final_image.select('B3', 'B2', 'B1'), {min:0, max:0.4}, 'RGB')
37
38
39 // uncomment the code below to export a image band to your drive
40 /*
41 Export.image.toDrive({
42   image: final_image.select('LST'),
43   description: 'Data_LST_2010',
44   scale: 30,
45   region: geometry,
46   fileFormat: 'GeoTIFF',
47 });
48 */
49
50 */
51 Reference :
52 Ermida, S.L., Soares, P., Mantas, V., Götsche, F.-M., Trigo, I.F., 2020.
53 | Google Earth Engine open-source code for Land Surface Temperature estimation from the Landsat series.
54 | Remote Sensing, 12 (9), 1471; https://doi.org/10.3390/rs12091471
55 */
```

2. Script penentuan LST tahun 2015

The screenshot shows the Google Earth Engine code editor interface. The title bar says "LST L8 single". The code itself is as follows:

```
LST L8 single
Get Link Save Run Reset Apps

Imports (1 entry)
var geometry: Polygon, 4 vertices

1 // link to the code that computes the Landsat LST
2 var LandsatLST = require('users/Mirnayani/GEE:Skripsi/Landsat_LST.js')
3
4
5 // select region of interest, date range, and landsat satellite
6 var satellite = 'L8';
7 var date_start = '2015-07-22';
8 var date_end = '2015-07-23';
9 var use_ndvi = true;
10
11 // get landsat collection with added variables: NDVI, FVC, TPW, EM, LST
12 var LandsatColl = LandsatLST.collection(satellite, date_start, date_end, geometry, use_ndvi)
13   .filter(ee.Filter.eq('WRS_PATH', 114))
14   .filter(ee.Filter.eq('WRS_ROW', 64))
15   .filter(ee.Filter.lt('CLOUD_COVER',10))
16
17 print(LandsatColl)
18
```

```

19 // select the first feature
20 var exImage = LandsatColl.first();
i 21 print(exImage)
22
23 var cmap1 = ['blue', 'green', 'yellow', 'orange', 'red'];
24 var cmap2 = ['F2F2F2','EFC2B3','ECB176','E9BD3A','E6E600','63C600','00A600'];
25
i 26 Map.centerObject(geometry, 10)
i 27 Map.addLayer(exImage.select('TPW'),{min:0.0, max:60.0, palette:cmap1},'TCW')
i 28 Map.addLayer(exImage.select('TPWpos'),{min:0.0, max:9.0, palette:cmap1},'TCWpos')
i 29 Map.addLayer(exImage.select('FVC'),{min:0.0, max:1.0, palette:cmap2}, 'FVC')
i 30 Map.addLayer(exImage.select('EM'),{min:0.9, max:1.0, palette:cmap1}, 'Emissivity')
i 31 Map.addLayer(exImage.select('B10'),{min:290, max:320, palette:cmap1}, 'TIR BT')
i 32 Map.addLayer(exImage.select('LST'),{min:290, max:320, palette:cmap1}, 'LST')
i 33 Map.addLayer(exImage.select('B4', 'B3', 'B2'), {min:0, max:0.4}, 'RGB')
i 33 Map.addLayer(exImage.select('B4', 'B3', 'B2'), {min:0, max:0.4}, 'RGB')
i 34
35 // uncomment the code below to export a image band to your drive
36
37 /*
38 Export.image.toDrive({
39   image: exImage.select('LST'),
40   description: 'Data_LST_2015',
41   scale: 30,
42   region: geometry,
43   fileFormat: 'GeoTIFF',
44 });
45 */
46
47 /*
48 Reference :
49 Ermida, S.L., Soares, P., Mantas, V., Götsche, F.-M., Trigo, I.F., 2020.
50 | Google Earth Engine open-source code for Land Surface Temperature estimation from the Landsat series.
51 | Remote Sensing, 12 (9), 1471; https://doi.org/10.3390/rs12091471
52 */

```

3. Script penentuan LST tahun 2020

```

LST L8 single
Get Link Save Run Reset Apps
Imports (1 entry) ↗
  var geometry: Polygon, 4 vertices ↗ ↘
1 // link to the code that computes the Landsat LST
i 2 var LandsatLST = require('users/Mirnayani/GEE:Skripsi/Landsat_LST.js')
3
4
5 // select region of interest, date range, and landsat satellite
6 var satellite = 'L8';
7 var date_start = '2020-08-20';
8 var date_end = '2020-08-21';
9 var use_ndvi = true;
10
11 // get landsat collection with added variables: NDVI, FVC, TPW, EM, LST
12 var LandsatColl = LandsatLST.collection(satellite, date_start, date_end, geometry, use_ndvi)
13   .filter(ee.Filter.eq('WRS_PATH', 114))
14   .filter(ee.Filter.eq('WRS_ROW', 64))
15   .filter(ee.Filter.lt('CLOUD_COVER',10))
16
i 17 print(LandsatColl)
18
19 // select the first feature
20 var exImage = LandsatColl.first();
i 21 print(exImage)
22
23 var cmap1 = ['blue', 'green', 'yellow', 'orange', 'red'];
24 var cmap2 = ['F2F2F2','EFC2B3','ECB176','E9BD3A','E6E600','63C600','00A600'];
25
i 26 Map.centerObject(geometry, 10)
i 27 Map.addLayer(exImage.select('TPW'),{min:0.0, max:60.0, palette:cmap1},'TCW')
i 28 Map.addLayer(exImage.select('TPWpos'),{min:0.0, max:9.0, palette:cmap1},'TCWpos')
i 29 Map.addLayer(exImage.select('FVC'),{min:0.0, max:1.0, palette:cmap2}, 'FVC')
i 30 Map.addLayer(exImage.select('EM'),{min:0.9, max:1.0, palette:cmap1}, 'Emissivity')
i 31 Map.addLayer(exImage.select('B10'),{min:290, max:320, palette:cmap1}, 'TIR BT')
i 32 Map.addLayer(exImage.select('LST'),{min:290, max:320, palette:cmap1}, 'LST')
i 33 Map.addLayer(exImage.select('B4', 'B3', 'B2'), {min:0, max:0.4}, 'RGB')
i 33 Map.addLayer(exImage.select('B4', 'B3', 'B2'), {min:0, max:0.4}, 'RGB')
i 34
35 // uncomment the code below to export a image band to your drive
36
37 /*
38 Export.image.toDrive({
39   image: exImage.select('LST'),
40   description: 'Data_LST_2020',
41   scale: 30,
42   region: geometry,
43   fileFormat: 'GeoTIFF',
44 });
45 */
46
47 /*

```

```

48     Reference :
49     Ermida, S.L., Soares, P., Mantas, V., Götsche, F.-M., Trigo, I.F., 2020.
50     | Google Earth Engine open-source code for Land Surface Temperature estimation from the Landsat series.
51     | Remote Sensing, 12 (9), 1471; https://doi.org/10.3390/rs12091471
52 */

```

4. Input script penentuan LST berdasarkan penelitian Ermida et al., 2020

```

Landsat_LST.js
Get Link Save Run Reset Apps
1  /*
2  Author: Sofia Ermida (sofia.ermida@ipma.pt; @ermida_sofia)
3
4 This code is free and open.
5 By using this code and any data derived with it,
6 you agree to cite the following reference
7 in any publications derived from them:
8 Ermida, S.L., Soares, P., Mantas, V., Götsche, F.-M., Trigo, I.F., 2020.
9     | Google Earth Engine open-source code for Land Surface Temperature estimation from the Landsat series.
10    | Remote Sensing, 12 (9), 1471; https://doi.org/10.3390/rs12091471
11
12 This function selects the Landsat data based on user inputs
13 and performs the LST computation
14
15 to call this function use:
16
17 var LandsatLST = require('users/sofiaermida/landsat_smw_lst:modules/Landsat_LST.js')
18 var LandsatCollection = LandsatLST.collection(landsat, date_start, date_end, geometry)
19
20 USES:
21     - NCEP_TPW.js
22     - cloudmask.js
23     - compute_NDVI.js
24     - compute_FVC.js
25     - compute_emissivity.js
26     - SMalgorithm.js
27
28 INPUTS:
29     - landsat: <string>
30         | the Landsat satellite id
31         | valid inputs: 'L4', 'L5', 'L7' and 'L8'
32     - date_start: <string>
33         | start date of the Landsat collection
34         | format: YYYY-MM-DD
35     - date_end: <string>
36         | end date of the Landsat collection
37         | format: YYYY-MM-DD
38     - geometry: <ee.Geometry>
39         | region of interest
40     - use_ndvi: <boolean>
41         | if true, NDVI values are used to obtain a
42         | dynamic emissivity; if false, emissivity is
43         | obtained directly from ASTER
44 OUTPUTS:
45     - <ee.ImageCollection>
46         | image collection with bands:
47         | landsat original bands: all from SR except the TIR bands (from TOA)
48         | - cloud masked
49         | - 'NDVI': normalized vegetation index
50         | - 'FVC': fraction of vegetation cover [0-1]
51         | - 'TPW': total precipitable water [mm]
52         | - 'EM': surface emissivity for TIR band
53         | - 'LST': land surface temperature
54
55     14-08-2020: update to avoid using the getInfo() and if()
56     (Thanks Tyler Erickson for the suggestion)
57 */
58
59 // MODULES DECLARATION -----
60 // Total Precipitable Water
i 61 var NCEP_TPW = require('users/sofiaermida/landsat_smw_lst:modules/NCEP_TPW.js')
i 62 //Cloud mask
i 63 var cloudmask = require('users/sofiaermida/landsat_smw_lst:modules/cloudmask.js')
i 64 //Normalized Difference Vegetation Index
i 65 var NDVI = require('users/sofiaermida/landsat_smw_lst:modules/compute_NDVI.js')
i 66 //Fraction of Vegetation cover
i 67 var FVC = require('users/sofiaermida/landsat_smw_lst:modules/compute_FVC.js')
i 68 //surface emissivity
i 69 var EM = require('users/sofiaermida/landsat_smw_lst:modules/compute_emissivity.js')
i 70 // land surface temperature
i 71 var LST = require('users/sofiaermida/landsat_smw_lst:modules/SMalgorithm.js')
i 72 // -----
73
74
75 var COLLECTION = ee.Dictionary({
76     'L7': {
77         'TOA': ee.ImageCollection('LANDSAT/LE07/C01/T1_TOA'),
78         'TIR': ['B6_VCID_1', 'B6_VCID_2'],
79     },
80     'L8': {
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
667
668
669
669
670
671
672
673
674
675
676
677
677
678
679
679
680
681
682
683
684
685
686
687
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
707
708
709
709
710
711
712
713
714
715
715
716
717
717
718
719
719
720
721
722
723
724
725
725
726
727
727
728
729
729
730
731
732
733
734
735
735
736
737
737
738
739
739
740
741
742
743
744
745
745
746
747
747
748
749
749
750
751
752
753
754
755
755
756
757
757
758
759
759
760
761
762
763
764
765
765
766
767
767
768
769
769
770
771
772
773
774
775
775
776
777
777
778
779
779
780
781
782
783
784
785
785
786
787
787
788
789
789
790
791
792
793
794
795
795
796
797
797
798
799
799
800
801
802
803
804
805
805
806
807
807
808
809
809
810
811
812
813
814
815
815
816
817
817
818
819
819
820
821
822
823
824
825
825
826
827
827
828
829
829
830
831
832
833
834
835
835
836
837
837
838
839
839
840
841
842
843
844
845
845
846
847
847
848
849
849
850
851
852
853
854
855
855
856
857
857
858
859
859
860
861
862
863
864
865
865
866
867
867
868
869
869
870
871
872
873
874
875
875
876
877
877
878
879
879
880
881
882
883
884
885
885
886
887
887
888
889
889
890
891
892
893
894
895
895
896
897
897
898
899
899
900
901
902
903
904
905
905
906
907
907
908
909
909
910
911
912
913
914
915
915
916
917
917
918
919
919
920
921
922
923
924
925
925
926
927
927
928
929
929
930
931
932
933
934
935
935
936
937
937
938
939
939
940
941
942
943
944
945
945
946
947
947
948
949
949
950
951
952
953
954
955
955
956
957
957
958
959
959
960
961
962
963
964
965
965
966
967
967
968
969
969
970
971
972
973
974
975
975
976
977
977
978
979
979
980
981
982
983
984
985
985
986
987
987
988
989
989
990
991
992
993
994
995
995
996
997
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1005
1006
1007
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1015
1016
1017
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1025
1026
1027
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1035
1036
1037
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1045
1046
1047
1047
1048
1049
1049
1050
1051
1052
1053
1054
1055
1055
1056
1057
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1065
1066
1067
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1075
1076
1077
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1085
1086
1087
1087
1088
1089
1089
1090
1091
1092
1093
1093
1094
1095
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1105
1106
1107
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1115
1116
1117
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1125
1126
1127
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1135
1136
1137
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1145
1146
1147
1147
1148
1149
1149
1150
1151
1152
1153
1154
1155
1155
1156
1157
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1165
1166
1167
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1175
1176
1177
1177
1178
1179
1179
1180
1181
1182
1183
1184
1185
1185
1186
1187
1187
1188
1189
1189
1190
1191
1192
1193
1194
1194
1195
1196
1196
1197
1198
1198
1199
1200
1201
1202
1203
1203
1204
1205
1205
1206
1207
1207
1208
1209
1209
1210
1211
1212
1213
1214
1214
1215
1216
1216
1217
1218
1218
1219
1220
1220
1221
1222
1223
1224
1225
1225
1226
1227
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1235
1236
1237
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1245
1246
1247
1247
1248
1249
1249
1250
1251
1252
1253
1254
1255
1255
1256
1257
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1265
1266
1267
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1275
1276
1277
1277
1278
1279
1279
1280
1281
1282
1283
1284
1285
1285
1286
1287
1287
1288
1289
1289
1290
1291
1292
1293
1294
1294
1295
1296
1296
1297
1298
1298
1299
1300
1301
1302
1303
1303
1304
1305
1305
1306
1307
1307
1308
1309
1309
1310
1311
1312
1313
1314
1314
1315
1316
1316
1317
1318
1318
1319
1320
1320
1321
1322
1323
1324
1325
1325
1326
1327
1327
1328
1329
1329
1330
1331
1332
1333
1334
1334
1335
1336
1336
1337
1338
1338
1339
1340
1340
1341
1342
1343
1344
1345
1345
1346
1347
1347
1348
1349
1349
1350
1351
1352
1353
1354
1355
1355
1356
1357
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1365
1366
1367
1367
1368
1369
1369
1370
1371
1372
1373
1374
1375
1375
1376
1377
1377
1378
1379
1379
1380
1381
1382
1383
1384
1385
1385
1386
1387
1387
1388
1389
1389
1390
1391
1392
1393
1394
1394
1395
1396
1396
1397
1398
1398
1399
1400
1401
1402
1403
1403
1404
1405
1405
1406
1407
1407
1408
1409
1409
1410
1411
1412
1413
1414
1414
1415
1416
1416
1417
1418
1418
1419
1420
1420
1421
1422
1423
1424
1425
1425
1426
1427
1427
1428
1429
1429
1430
1431
1432
1433
1434
1434
1435
1436
1436
1437
1438
1438
1439
1440
1440
1441
1442
1443
1444
1445
1445
1446
1447
1447
1448
1449
1449
1450
1451
1452
1453
1454
1455
1455
1456
1457
1457
1458
1459
1459
1460
1461
1462
1463
1464
1464
1465
1466
1466
1467
1468
1468
1469
1470
1470
1471
1472
1473
1474
1475
1475
1476
1477
1477
1478
1479
1479
1480
1481
1482
1483
1484
1484
1485
1486
1486
1487
1488
1488
1489
1490
1490
1491
1492
1493
1494
1495
1495
1496
1497
1497
1498
1499
1499
1500
1501
1502
1503
1504
1504
1505
1506
1506
1507
1508
1508
1509
1510
1511
1512
1513
1513
1514
1515
1515
1516
1517
1517
1518
1519
1519
1520
1521
1522
1523
1524
1524
1525
1526
1526
1527
1528
1528
1529
1530
1531
1532
1533
1534
1534
1535
1536
1536
1537
1538
1538
1539
1540
1541
1542
1543
1544
1544
1545
1546
1546
1547
1548
1548
1549
1550
1551
1552
1553
1554
1554
1555
1556
1556
1557
1558
1558
1559
1560
1561
1562
1563
1564
1564
1565
1566
1566
1567
1568
1568
1569
1570
1571
1572
1573
1574
1574
1575
1576
1576
1577
1578
1578
1579
1580
1581
1582
1583
1584
1584
1585
1586
1586
1587
1588
1588
1589
1590
1591
1592
1593
1594
1594
1595
1596
1596
1597
1598
1598
1599
1600
1601
1602
1603
1603
1604
1605
1605
1606
1607
1607
1608
1609
1609
1610
1611
1612
1613
1614
1614
1615
1616
1616
1617
1618
1618
1619
1620
1620
1621
1622
1623
1624
1625
1625
1626
1627
1627
1628
1629
1629
1630
1631
1632
1633
1634
1634
1635
1636
1636
1637
1638
1638
1639
1640
1640
1641
1642
1643
1644
1645
1645
1646
1647
1647
1648
1649
1649
1650
1651
1652
1653
1654
1655
1655
1656
1657
1657
1658
1659
1659
1660
1661
1662
1663
1664
1664
1665
1666
1666
1667
1668
1668
1669
1670
1670
1671
1672
1673
1674
1675
1675
1676
1677
1677
1678
1679
1679
1680
1681
1682
1683
1684
1684
1685
1686
1686
1687
1688
1688
1689
1690
1690
1691
1692
1693
1694
1695
1695
1696
1697
1697
1698
1699
1699
1700
1701
1702
1703
1704
1704
1705
1706
1706
1707
1708
1708
1709
1710
1711
1712
1713
1713
1714
1715
1715
1716
1717
1717
1718
1719
1719
1720
1721
1722
1723
1724
1724
1725
1726
1726
1727
1728
1728
1729
1730
1731
1732
1733
1734
1734
1735
1736
1736
1737
1738
1738
1739
1740
1741
1742
1743
1744
1744
1745
1746
1746
1747
1748
1748
1749
1750
1751
1752
1753
1754
1754
1755
1756
1756
1757
1758
1758
1759
1760
1761
1762
1763
1764
1764
1765
1766
1766
1767
1768
1768
1769
1770
1771
1772
1773
1774
1774
1775
1776
1776
1777
1778
1778
1779
1780
1781
1782
1783
1784
1784
1785
1786
1786
1787
1788
1788
1789
1790
1791
1792
1793
1794
1794
1795
1796
1796
1797
1798
1798
1799
1800
1801
1802
1803
1803
1804
1805
1805
1806
1807
1807
1808
1809
1809
1810
1811
1812
1813
1814
1814
1815
1816
1816
1817
1818
1818
1819
1820
1821
1822
1823
1824
1824
1825
1826
1826
1827
1828
1828
1829
1830
1831
1832
1833
1834
1834
1835
1836
1836
1837
1838
1838
1839
1840
1841
1842
1843
1844
1844
1845
1846
1846
1847
1848
1848
1849
1850
1851
1852
1853
1854
1854
1855
1856
1856
1857
1858
1858
1859
1860
1861
1862
1863
18
```

```

81     'TOA': ee.ImageCollection('LANDSAT/LC08/C01/T1_TOA'),
82     'TIR': ['B10','B11']
83   });
84 });
85
86
87 // exports.collection = function(landsat, date_start, date_end, geometry, use_ndvi) {
88
89 // load TOA Radiance/Reflectance
90 var collection_dict = ee.Dictionary(COLLECTION.get(landsat));
91
92 var landsatTOA = ee.ImageCollection(collection_dict.get('TOA'))
93   .filter(ee.Filter.date(date_start, date_end))
94   .filterBounds(geometry)
95   .map(cloudmask.toa)
96   .map(NDVI.addBand(landsat))
97   .map(FVC.addBand(landsat))
98   .map(NCEP_TPW.addBand)
99   .map(EM.addBand(landsat,use_ndvi));
100
101 // compute the LST
102 var tir = ee.List(collection_dict.get('TIR'))
103 var landsatLST = landsatTOA.map(LST.addBand(landsat));
104
105 return landsatLST;
106 };

```

NCEP_TPW.js

Get Link Save Run Reset Apps

```

1 /*
2 Author: Sofia Ermida (sofia.ermida@ipma.pt; @ermida_sofia)
3
4 This code is free and open.
5 By using this code and any data derived with it,
6 you agree to cite the following reference
7 in any publications derived from them:
8 Ermida, S.L., Soares, P., Mantas, V., Götsche, F.-M., Trigo, I.F., 2020.
9 Google Earth Engine open-source code for Land Surface Temperature estimation from the Landsat series.
10 Remote Sensing, 12 (9), 1471; https://doi.org/10.3390/rs12091471
11
12 this function matches the atmospheric water vapour data
13 from NCEP reanalysis to each Landsat image
14 tpw values are interpolated from the 6-hourly model times to the image time
15
16 to call this function use:
17
18 var NCEP_TPW = require('users/sofiaermida/landsat_smw_lst:modules/NCEP_TPW.js')
19 var ImagewithTPW = NCEP_TPW.addBand(image)
20 or
21 var collectionwithPTW = ImageCollection.map(NCEP_TPW.addBand)
22
23 INPUTS:
24   - image: <ee.Image>
25     image for which to interpolate the TPW data
26     needs the 'system:time_start' image property
27 OUTPUTS:
28   - <ee.Image>
29     the input image with 3 new bands:
30     'TPW': total precipitable water values
31     'TPWpos': index for the LUT of SMW algorithm coefficients
32
33 10.12.2020: typo correction in the tpw inperpolation expression
34  (thanks to Jiacheng Zhao for reporting this issue)
35 */
36
37 exports.addBand = function(image){
38
39 // first select the day of interest
40 var date = ee.Date(image.get('system:time_start'))
41 var year = ee.Number.parse(date.format('yyyy'))
42 var month = ee.Number.parse(date.format('MM'))
43 var day = ee.Number.parse(date.format('dd'))
44 var date1 = ee.Date.fromMD(year,month,day)
45 var date2 = date1.advance(1,'days')
46
47 // function compute the time difference from landsat image
48 var datedist = function(image){
49   return image.set('DateDist',
50     ee.Number(image.get('system:time_start'))
51     .subtract(date.millis()).abs())
52 }
53
54 // load atmospheric data collection
55 var TPWcollection = ee.ImageCollection('NCEP_RE/surface_wv')
56   .filter(ee.Filter.date(date1.format('yyyy-MM-dd'), date2.format('yyyy-MM-dd')))
57   .map(datedist)
58
59 // select the two closest model times
60 var closest = (TPWcollection.sort('DateDist')).toList(2);

```

```

61 // check if there is atmospheric data in the wanted day
62 // if not creates a TPW image with non-realistic values
63 // these are then masked in the SMWalgorithm function (prevents errors)
64 var tpw1 = ee.Image(ee.Algorithms.If(closest.size().eq(0), ee.Image.constant(-999.0),
65   ee.Image(closest.get(0)).select('pr_wtr')));
66 var tpw2 = ee.Image(ee.Algorithms.If(closest.size().eq(0), ee.Image.constant(-999.0),
67   ee.Algorithms.If(closest.size().eq(1), tpw1,
68     ee.Image(closest.get(1)).select('pr_wtr'))));
69
70
71 var time1 = ee.Number(ee.Algorithms.If(closest.size().eq(0), 1.0,
72   ee.Number(tpw1.get('DateDist')).divide(ee.Number(21600000)) ));
73 var time2 = ee.Number(ee.Algorithms.If(closest.size().lt(2), 0.0,
74   ee.Number(tpw2.get('DateDist')).divide(ee.Number(21600000)) ));
75
76 var tpw = tpw1.expression('tpw1*time2+tpw2*time1',
77   {'tpw1':tpw1,
78    'time1':time1,
79    'tpw2':tpw2,
80    'time2':time2
81   }).clip(image.geometry());
82
83 // SMW coefficients are binned by TPW values
84 // find the bin of each TPW value
85 var pos = tpw.expression(
86   "value = (TPW>0 && TPW<=6) ? 0" +
87   ": (TPW6 && TPW<=12) ? 1" +
88   ": (TPW12 && TPW<=18) ? 2" +
89   ": (TPW>18 && TPW<=24) ? 3" +
90   ": (TPW24 && TPW<=30) ? 4" +
91   ": (TPW30 && TPW<=36) ? 5" +
92   ": (TPW36 && TPW<=42) ? 6" +
93   ": (TPW42 && TPW<=48) ? 7" +
94   ": (TPW48 && TPW<=54) ? 8" +
95   ": (TPW>54) ? 9" +
96   ": 0",{'TPW': tpw})
97 .clip(image.geometry());
98
99 // add tpw to image as a band
100 var withTPW = (image.addBands(tpw.rename('TPW'),['TPW'])).addBands(pos.rename('TPWpos'),['TPWpos']);
101
102 return withTPW
103 };

```

SMW_coefficients.js

Get Link | Save | Run | Reset | Apps

```

1 /*
2 Author: Sofia Ermida (sofia.ermida@ipma.pt; @ermida_sofia)
3
4 This code is free and open.
5 By using this code and any data derived with it,
6 you agree to cite the following reference
7 in any publications derived from them:
8 Ermida, S.L., Soares, P., Mantas, V., Götsche, F.-M., Trigo, I.F., 2020.
9 | Google Earth Engine open-source code for Land Surface Temperature estimation from the Landsat series.
10 | Remote Sensing, 12 (9), 1471; https://doi.org/10.3390/rs12091471
11 */
12 // coefficients for the Statistical Mono-Window Algorithm
13 exports.coeff_SMW_L4 = ee.FeatureCollection([
14   ee.Feature(null, {'TPWpos': 0, 'A': 0.9755, 'B': -205.2767, 'C': 212.0051}),
15   ee.Feature(null, {'TPWpos': 1, 'A': 1.0155, 'B': -233.8902, 'C': 230.4049}),
16   ee.Feature(null, {'TPWpos': 2, 'A': 1.0672, 'B': -257.1884, 'C': 239.3072}),
17   ee.Feature(null, {'TPWpos': 3, 'A': 1.1499, 'B': -286.2166, 'C': 244.8497}),
18   ee.Feature(null, {'TPWpos': 4, 'A': 1.2277, 'B': -316.7643, 'C': 253.0033}),
19   ee.Feature(null, {'TPWpos': 5, 'A': 1.3649, 'B': -361.8276, 'C': 258.5471}),
20   ee.Feature(null, {'TPWpos': 6, 'A': 1.5085, 'B': -410.1157, 'C': 265.1111}),
21   ee.Feature(null, {'TPWpos': 7, 'A': 1.7045, 'B': -472.4909, 'C': 270.7000}),
22   ee.Feature(null, {'TPWpos': 8, 'A': 1.5886, 'B': -442.9489, 'C': 277.1511}),
23   ee.Feature(null, {'TPWpos': 9, 'A': 2.0215, 'B': -571.8563, 'C': 279.9854})
24 ]);
25
26 exports.coeff_SMW_L5 = ee.FeatureCollection([
27   ee.Feature(null, {'TPWpos': 0, 'A': 0.9765, 'B': -204.6584, 'C': 211.1321}),
28   ee.Feature(null, {'TPWpos': 1, 'A': 1.0229, 'B': -235.5384, 'C': 230.0619}),
29   ee.Feature(null, {'TPWpos': 2, 'A': 1.0817, 'B': -261.3886, 'C': 239.5256}),
30   ee.Feature(null, {'TPWpos': 3, 'A': 1.1738, 'B': -293.6128, 'C': 245.6042}),
31   ee.Feature(null, {'TPWpos': 4, 'A': 1.2605, 'B': -327.1417, 'C': 254.2301}),
32   ee.Feature(null, {'TPWpos': 5, 'A': 1.4166, 'B': -377.7741, 'C': 259.9711}),
33   ee.Feature(null, {'TPWpos': 6, 'A': 1.5727, 'B': -430.0388, 'C': 266.9520}),
34   ee.Feature(null, {'TPWpos': 7, 'A': 1.7879, 'B': -498.1947, 'C': 272.8413}),
35   ee.Feature(null, {'TPWpos': 8, 'A': 1.6347, 'B': -457.8183, 'C': 279.6160}),
36   ee.Feature(null, {'TPWpos': 9, 'A': 2.1168, 'B': -600.7079, 'C': 282.4583})
37 ]);
38
39 exports.coeff_SMW_L7 = ee.FeatureCollection([
40   ee.Feature(null, {'TPWpos': 0, 'A': 0.9764, 'B': -205.3511, 'C': 211.8507}),
41   ee.Feature(null, {'TPWpos': 1, 'A': 1.0201, 'B': -235.2416, 'C': 230.5468}),
42   ee.Feature(null, {'TPWpos': 2, 'A': 1.0750, 'B': -259.6560, 'C': 239.6619}),
43   ee.Feature(null, {'TPWpos': 3, 'A': 1.1612, 'B': -289.8190, 'C': 245.3286}),
44   ee.Feature(null, {'TPWpos': 4, 'A': 1.2425, 'B': -321.4658, 'C': 253.6144})
45 ]);

```



```

71           ee.Algorithms.If(landsat=='L7','B6_VCID_1',
72                     'B6')));
73     // compute the LST
74     var lst = image.expression(
75       'A*Tb1/em1 + B/em1 + C',
76       {'A': A_img,
77        'B': B_img,
78        'C': C_img,
79        'em1': image.select('EM'),
80        'Tb1': image.select(tir)
81      }).updateMask(image.select('TPW').lt(0).not());
82
83
84     return image.addBands(lst.rename('LST'))
85   );
86   return wrap
87 }

```

compute_FVC.js

```

15 to call this function use:
16
17 var FVCfun = require('users/sofiaermida/landsat_smw_lst:modules/compute_FVC.js')
18 var ImagewithFVC = FVCfun.addBand(landsat)(image)
19 or
20 var collectionwithFVC = ImageCollection.map(FVCfun.addBand(landsat))
21
22 USES:
23 | - SMW_coefficients.js
24
25 INPUTS:
26 | - landsat: <string>
27 |       the Landsat satellite id
28 |       valid inputs: 'L4', 'L5', 'L7' and 'L8'
29 | - image: <ee.Image>
30 |       image for which to calculate the FVC
31 OUTPUTS:
32 | - <ee.Image>
33 |       the input image with 1 new band:
34 |       'FVC': fraction of vegetation cover
35 */
36 exports.addBand = function(landsat){
37   var wrap = function(image){
38
39     var ndvi = image.select('NDVI')
40
41     // Compute FVC
42     var fvc = image.expression('((ndvi-ndvi_bg)/(ndvi_vg - ndvi_bg))**2',
43       {'ndvi':ndvi,'ndvi_bg':0.2,'ndvi_vg':0.86});
44     fvc = fvc.where(fvc.lt(0.0),0.0);
45     fvc = fvc.where(fvc.gt(1.0),1.0);
46
47     return image.addBands(fvc.rename('FVC'));
48   }
49   return wrap
50 };

```

compute_NDVI.js

```

5
6
7 to call this function use:
8
9 var NDVIfun = require('users/sofiaermida/landsat_smw_lst:modules/compute_NDVI.js')
10 var ImagewithNDVI = NDVIfun.addBand(landsat)(image)
11 or
12 var collectionwithNDVI = ImageCollection.map(NDVIfun.addBand(landsat))
13
14 INPUTS:
15 | - landsat: <string>
16 |       the Landsat satellite id
17 |       valid inputs: 'L4', 'L5', 'L7' and 'L8'
18 | - image: <ee.Image>
19 |       image for which to calculate the NDVI
20 OUTPUTS:
21 | - <ee.Image>
22 |       the input image with 1 new band:
23 |       'NDVI': normalized difference vegetation index
24 */
25
26 exports.addBand = function(landsat){
27   var wrap = function(image){
28
29     // choose bands
30     var nir = ee.String(ee.Algorithms.If(landsat=='L8','B5','B4'))
31     var red = ee.String(ee.Algorithms.If(landsat=='L8','B4','B3'))
32
33     // compute NDVI
34     return image.addBands(image.expression('(nir-red)/(nir+red)',{
35       'nir':image.select(nir).multiply(0.0001),
36       'red':image.select(red).multiply(0.0001)
37     });
38   }
39   return wrap
40 };

```

```

i 37 |     }).rename('NDVI'))
i 38 }
i 39     return wrap
i 40 };

compute_emissivity.js
Get Link Save Run Reset Apps
43 // EM: surface emissivity of TIR band
44 */
45
i 46 var ASTERGED = require('users/sofiaermida/landsat_smw_lst:modules/ASTER_bare_emiss.js')
47
48 // this function computes the emissivity of the
49 // Landsat TIR band using ASTER and FVC
50 exports.addBand = function(landsat, use_ndvi){
51     var wrap = function(image){
52
53         var c13 = ee.Number(ee.Algorithms.If(landsat==='L4',0.3222,
54                                         ee.Algorithms.If(landsat==='L5',-0.0723,
55                                         ee.Algorithms.If(landsat==='L7',0.2147,
56                                         0.6820)));
57         var c14 = ee.Number(ee.Algorithms.If(landsat==='L4',0.6498,
58                                         ee.Algorithms.If(landsat==='L5',1.0521,
59                                         ee.Algorithms.If(landsat==='L7',0.7789,
60                                         0.2578)));
61         var c = ee.Number(ee.Algorithms.If(landsat==='L4',0.0272,
62                                         ee.Algorithms.If(landsat==='L5',0.0195,
63                                         ee.Algorithms.If(landsat==='L7',0.0059,
64                                         0.0584)));
65
66         // get ASTER emissivity
67         // convolve to Landsat band
68         var emiss_bare = image.expression('c13*EM13 + c14*EM14 + c',{
69             'EM13':ASTERGED.emiss_bare_band13(image),
70             'EM14':ASTERGED.emiss_bare_band14(image),
71             'c13':ee.Image(c13),
72             'c14':ee.Image(c14),
73             'c':ee.Image(c)
74         });
75
76         // compute the dynamic emissivity for Landsat
77         var EMd = image.expression('fvc*0.99+(1-fvc)*em_bare',
78             {'fvc':image.select('FVC'),'em_bare':emiss_bare});
79
80         // compute emissivity directly from ASTER
81         // without vegetation correction
82         // get ASTER emissivity
83         var aster = ee.Image("NASA/ASTER_GED/AG100_003")
84             .clip(image.geometry());
85         var EM0 = image.expression('c13*EM13 + c14*EM14 + c',{
86             'EM13':aster.select('emissivity_band13').multiply(0.001),
87             'EM14':aster.select('emissivity_band14').multiply(0.001),
88             'c13':ee.Image(c13),
89             'c14':ee.Image(c14),
90             'c':ee.Image(c)
91         });
92
93         // select which emissivity to output based on user selection
94         var EM = ee.Image(ee.Algorithms.If(use_ndvi,EMd,EM0));
95
96         return image.addBands(EM.rename('EM'));
97     }
98     return wrap
i 99 }

```

5. Script pengolahan NDBI, NDVI, NDWI tahun 2010

```

Predictor LST L7
Get Link Save Run Reset Apps
    ▾ Imports (1 entry) ▾
        ▾ var geometry: Table users/Mirnayani/shpmakassar
1 //load Landsat 7 Tier 1 TOA image collection
2 var MIN_SCALE = 1/3;
3 var MAX_SCALE = 3;
4 var MIN_NEIGHBORS = 144;
5
6 /* Apply the USGS L7 Phase-2 Gap filling protocol, using a single kernel size. */
7 var GapFill = function(src, fill, kernelSize, upscale) {
i 8     var kernel = ee.Kernel.square(kernelSize * 30, "meters", false)
9
10    // Find the pixels common to both scenes.
11    var common = src.mask().and(fill.mask())
12    var fc = fill.updateMask(common)
13    var sc = src.updateMask(common)
14    Map.addLayer(common.select(0).mask(common.select(0)), {palette:['000000']}, 'common mask (both exist)', false)
15
16    // Find the primary scaling factors with a regression.
17    // Interleave the bands for the regression. This assumes the bands have the same names.
18    var regress = fc.addBands(sc)
19
20    regress = regress.select(regress.bandNames().sort())
21

```

```

i 22  var ratio = 5
i 23
i 24  if(upscale) {
i 25    var fit = regress
i 26      .reduceResolution(ee.Reducer.median(), false, 500)
i 27      .reproject(regress.select(0).projection().scale(ratio, ratio))
i 28      .reduceNeighborhood(ee.Reducer.linearFit().forEach(src.bandNames()), kernel, null, false)
i 29      .unmask()
i 30      .reproject(regress.select(0).projection().scale(ratio, ratio))
i 31  } else {
i 32    var fit = regress
i 33      .reduceNeighborhood(ee.Reducer.linearFit().forEach(src.bandNames()), kernel, null, false)
i 34  }
i 35
i 36  var offset = fit.select(".*_offset")
i 37  var scale = fit.select(".*_scale")
i 38
i 39  Map.addLayer(scale.select('B1_scale'), {min:-2, max:2}, 'scale B1', false)
i 40
i 41 // Find the secondary scaling factors using just means and stddev
i 42 var reducer = ee.Reducer.mean().combine(ee.Reducer.stdDev(), null, true)
i 43
i 44  if(upscale) {
i 45    var src_stats = src
i 46      .reduceResolution(ee.Reducer.median(), false, 500)
i 47      .reproject(regress.select(0).projection().scale(ratio, ratio))
i 48      .reduceNeighborhood(reducer, kernel, null, false)
i 49      .reproject(regress.select(0).projection().scale(ratio, ratio))
i 50
i 51    var fill_stats = fill
i 52      .reduceResolution(ee.Reducer.median(), false, 500)
i 53      .reproject(regress.select(0).projection().scale(ratio, ratio))
i 54      .reduceNeighborhood(reducer, kernel, null, false)
i 55      .reproject(regress.select(0).projection().scale(ratio, ratio))
i 56  } else {
i 57    var src_stats = src
i 58      .reduceNeighborhood(reducer, kernel, null, false)
i 59
i 60    var fill_stats = fill
i 61      .reduceNeighborhood(reducer, kernel, null, false)
i 62  }
i 63
i 64  var scale2 = src_stats.select(".*stdDev").divide(fill_stats.select(".*stdDev"))
i 65  var offset2 = src_stats.select(".*mean").subtract(fill_stats.select(".*mean").multiply(scale2))
i 66
i 67  var invalid = scale.lt(MIN_SCALE).or(scale.gt(MAX_SCALE))
i 68  Map.addLayer(invalid.select(0).mask(invalid.select(0)), {palette:['550000']}, 'invalid1', false)
i 69  scale = scale.where(invalid, scale2)
i 70  offset = offset.where(invalid, offset2)
i 71
i 72 // When all else fails, just use the difference of means as an offset.
i 73  var invalid2 = scale.lt(MIN_SCALE).or(scale.gt(MAX_SCALE))
i 74  Map.addLayer(invalid2.select(0).mask(invalid2.select(0)), {palette:['552020']}, 'invalid2', false)
i 75  scale = scale.where(invalid2, 1)
i 76  offset = offset.where(invalid2, src_stats.select(".*mean").subtract(fill_stats.select(".*mean")))
i 77
i 78 // Apply the scaling and mask off pixels that didn't have enough neighbors.
i 79  var count = common.reduceNeighborhood(ee.Reducer.count(), kernel, null, true, "boxcar")
i 80  var scaled = fill.multiply(scale).add(offset)
i 81  ||| .updateMask(count.gte(MIN_NEIGHBORS))
i 82
i 83  return src.unmask(scaled, true)
i 84
i 85
i 86  var source = ee.Image("LANDSAT/LE07/C01/T1_TOA/LE07_114064_20100902")
i 87  var fill = ee.Image("LANDSAT/LE07/C01/T1_TOA/LE07_114064_20100310")
i 88
i 89  /*
i 90  Map.addLayer(source, {bands: ['B3', 'B2', 'B1'], min: 0, max: 0.3}, 'Makassar_2020src');
i 91  Map.addLayer(fill, {bands: ['B3', 'B2', 'B1'], min: 0, max: 0.3}, 'Makassar_2020fill');
i 92 */
i 93
i 94  /*
i 95  Map.addLayer(fill, {min:0, max:0.3, bands:["B3", "B2", "B1"]}, "fill", true)
i 96  Map.addLayer(source, {min:0, max:0.3, bands:["B3", "B2", "B1"]}, "destination", true)
i 97 */
i 98
i 99  var L7 = GapFill(source, fill, 10, false);
i 100 Map.addLayer(L7, {min:0, max:0.3, bands:["B3", "B2", "B1"]}, "filled", true)
i 101
i 102 var L7 = GapFill(source, fill, 10, true);
i 103 Map.addLayer(L7, {min:0, max:0.3, bands:["B3", "B2", "B1"]}, "filled (upscaled)", true)
i 104
i 105
i 106
i 107 //Display the clipped image collection with visual parameters
i 108 Map.addLayer(L7, {bands: ['B3', 'B2', 'B1'], min: 0, max: 0.3}, 'Makassar_2010');
i 109

```

```

110
111 // Compute the Normalized Difference Vegetation Index (NDVI).
112 var addNDVI = function(L7) {
113   var ndvi = L7.normalizedDifference(['B4', 'B3']).rename('NDVI');
114   return L7.addBands(ndvi);
115 };
116
117 // Test the addNDVI function on a single image.
118 var ndvi = addNDVI(L7).select('NDVI');
119
120 // Display the result.
121 Map.centerObject(geometry, 9);
122 var ndviParams = {min: -1, max: 1, palette: ['blue', 'white', 'green']};
123 Map.addLayer(ndvi, ndviParams, 'NDVI image');
124
125 // Compute the Normalized Difference Water Index (NDWI).
126 var addNDWI = function(L7) {
127   var ndwi = L7.normalizedDifference(['B4', 'B5']).rename('NDWI');
128   return L7.addBands(ndwi);
129 };
130
131 // Test the addNDWI function on a single image.
132 var ndwi = addNDWI(L7).select('NDWI');
133
134 // Display the result.
135 Map.centerObject(geometry, 9);
136 var ndwiParams = {min: 0.5, max: 1, palette: ['00FFFF', '0000FF']};
137 Map.addLayer(ndwi, ndwiParams, 'NDWI image');
138
139 // Compute the Normalized Difference Built-up Index (NDBI).
140 var addNDBI = function(L7) {
141   var ndbi = L7.normalizedDifference(['B5','B4']).rename('NDBI');
142   return L7.addBands(ndbi);
143 };
144
145 // Test the addNDBI function on a single image.
146 var ndbi = addNDBI(L7).select('NDBI');
147
148 // Display the result.
149 Map.centerObject(geometry, 9);
150 var ndbiParams = {min: -1, max: 1, palette: ['white', 'orange', 'red']};
151 Map.addLayer(ndbi, ndbiParams, 'NDBI image');
152
153 var stacked_L7 = L7.addBands(ndvi).addBands(ndwi).addBands(ndbi);
154 print('stacked_L7', stacked_L7.bandNames());
155
156 //Display the stacked_composite
157 Map.addLayer(stacked_L7, {bands: ['B3', 'B2', 'B1'], min: 0, max: 0.3}, 'Makassar2010_stack');
158
159 /*
160 Export.image.toDrive({
161   image: stacked_L7.select('NDVI'),
162   description: 'Data_2010_NDVI',
163   scale: 30,
164   region: geometry,
165   fileFormat: 'GeoTIFF',
166 });
167
168 */
169 Export.image.toDrive({
170   image: stacked_L7.select('NDBI'),
171   description: 'Data_2010_NDBI',
172   scale: 30,
173   region: geometry,
174   fileFormat: 'GeoTIFF',
175 });
176
177 Export.image.toDrive({
178   image: stacked_L7.select('NDWI'),
179   description: 'Data_2010_NDWI',
180   scale: 30,
181   region: geometry,
182   fileFormat: 'GeoTIFF',
183 });
184 */

```

6. Data Suhu Udara Harian dari BMKG



ID WMO	:	97182
Nama Stasiun	:	Stasiun Meteorologi Maritim Paotere
Lintang	:	-5.11375
Bujur	:	119.41983
Elevasi	:	5

Tanggal	Tn	Tx	Tavg
02-09-2010	24,8	32,8	28,5
22-07-2015	23,8	32,2	27
20-08-2020	23,5	32	27,4

Keterangan :

Tn: Temperatur minimum (°C)

Tx: Temperatur maksimum (°C)

Tavg: Temperatur rata-rata (°C)

7. Data Suhu Udara Tahunan dari BMKG Wilayah IV Makassar

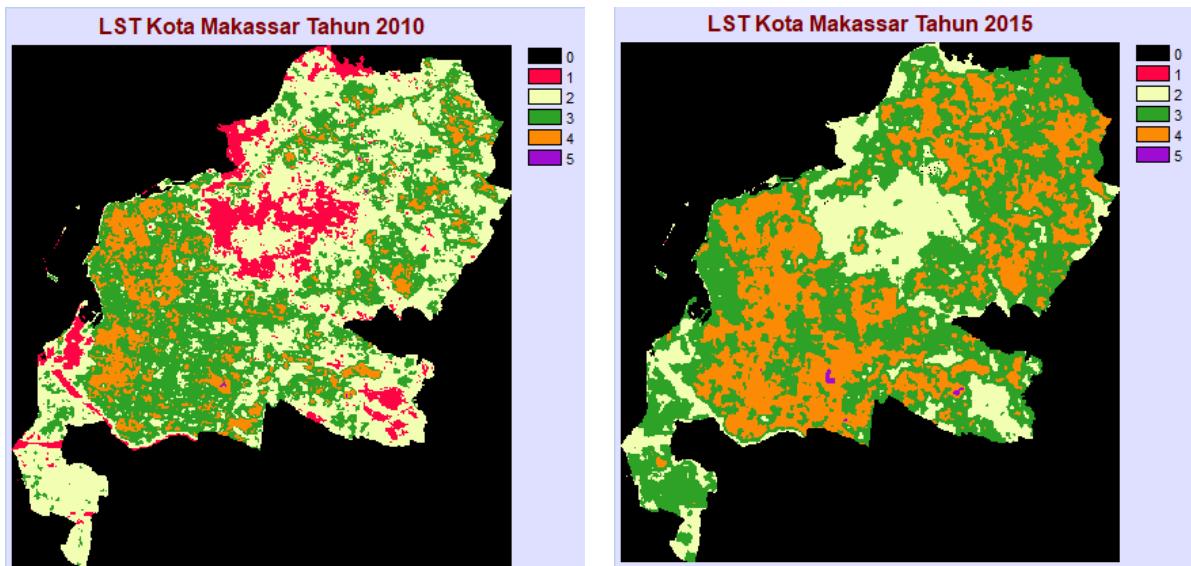
DATA SUHU UDARA STASIUN METEOROLOGI MARITIM PAOTERE, KOTA MAKASSAR, SULAWESI SELATAN

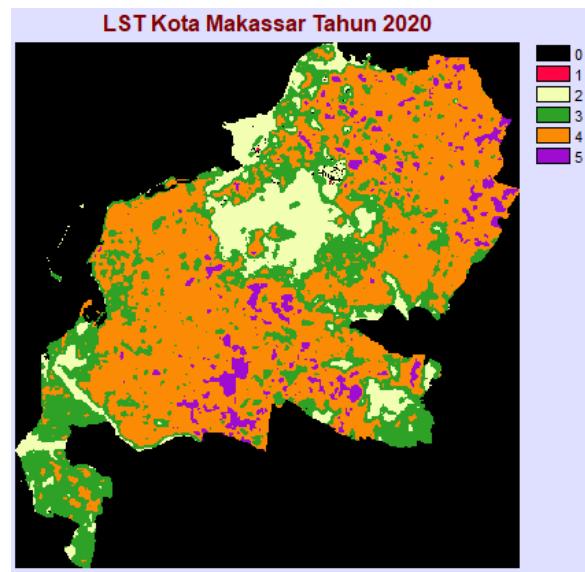
KOORDINAT : 05° 06' 49.5" LS, 119° 25' 11.4" BT
TINGGI : 1.75 m

SUHU UDARA RATA RATA (dalam derajat celcius)

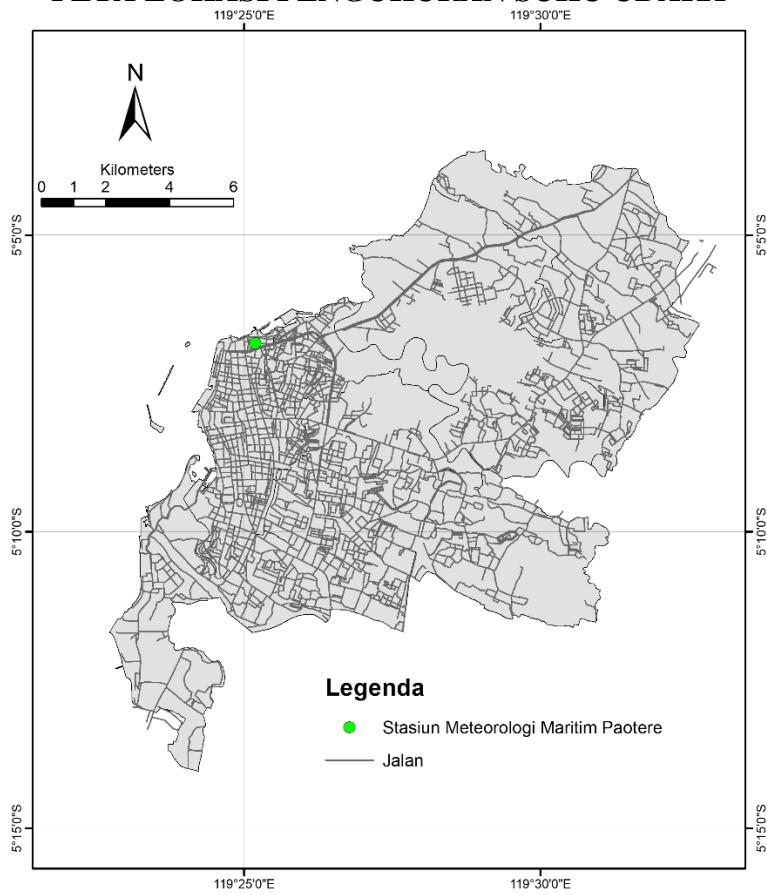
BULAN TAHUN	JAN	FEB	MAR	APR	MEI	JUN	JUL	AGS	SEP	OKT	NOV	DES
2010	26.6	27.8	28.2	28.5	28.5	28.1	27.8	28.1	28.0	28.1	28.2	26.6
2015	26.8	27.2	27.5	27.8	28.3	27.7	27.4	27.3	27.6	29.0	29.4	27.9
2020	28.3	27.8	28.3	28.5	29.0	28.4	28.2	28.6	29.1	29.3	28.9	27.0

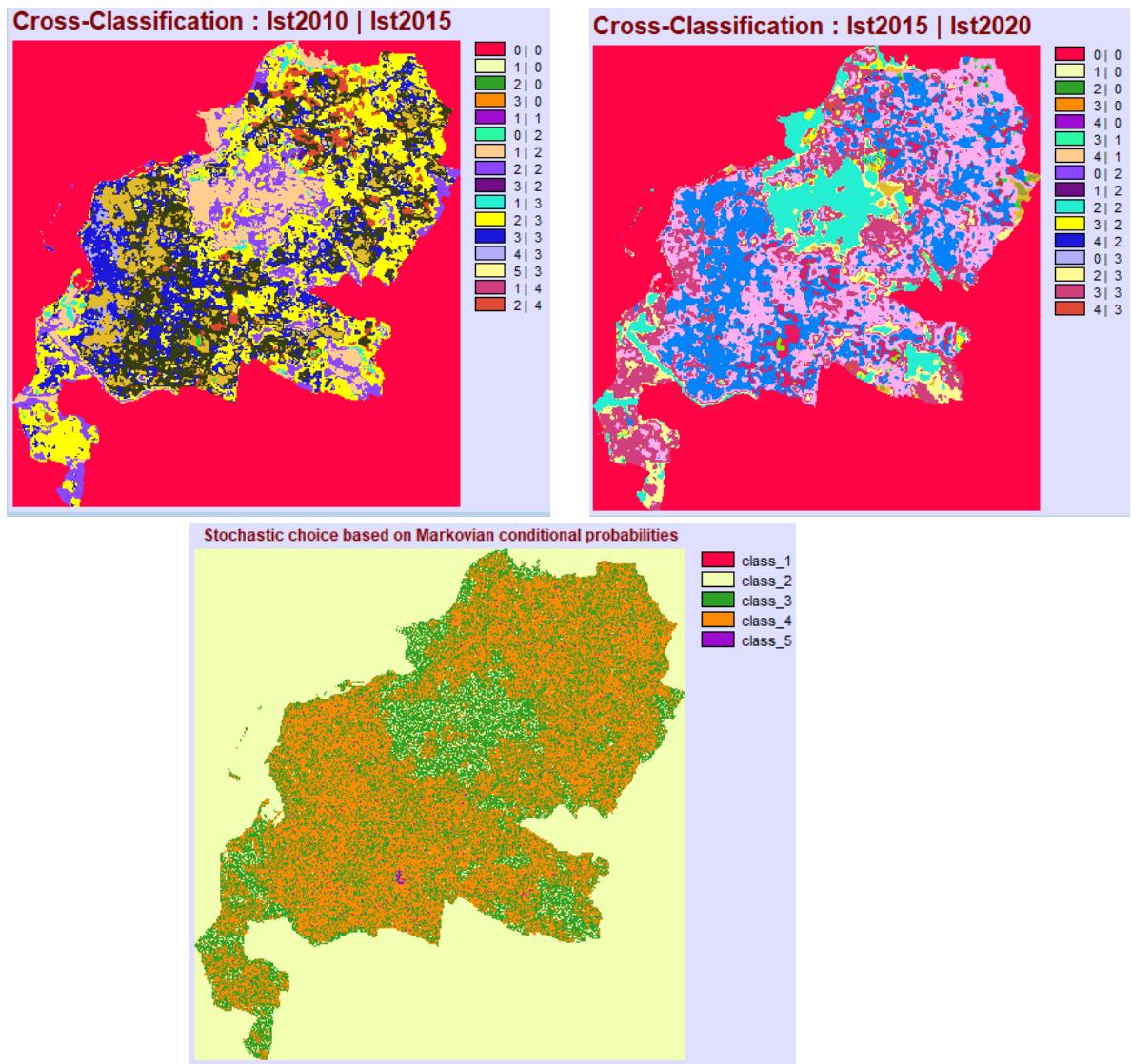
8. Pengolahan Model Simulasi Prediksi LST tahun 2025



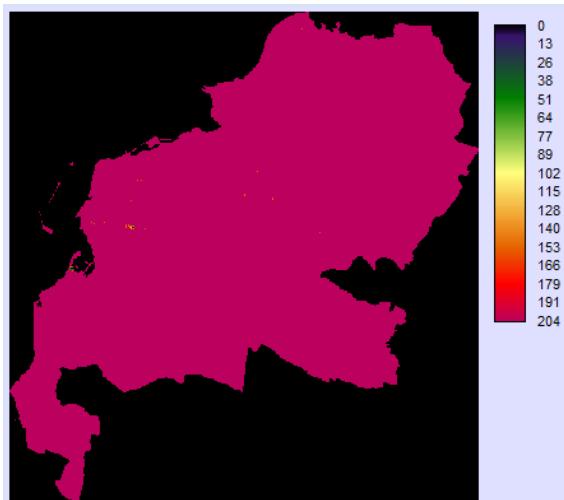


PETA LOKASI PENGUKURAN SUHU UDARA

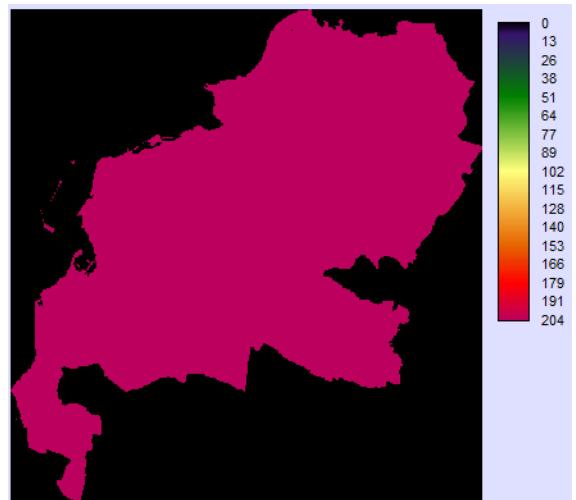




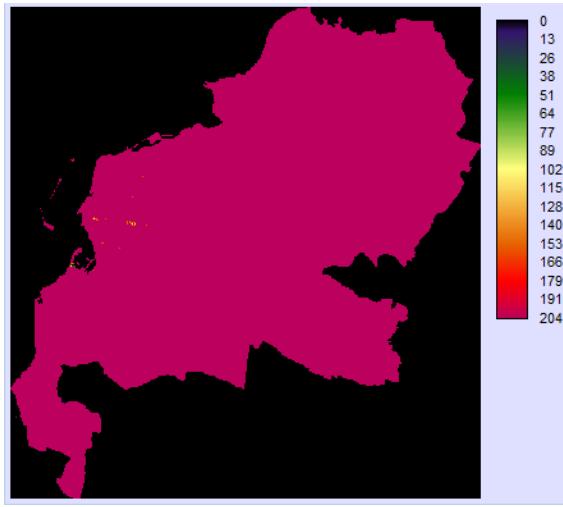
Faktor Pendorong : NDBI tahun 2010



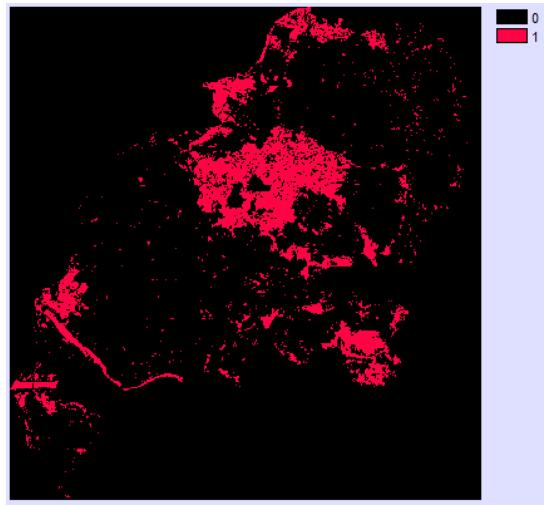
Faktor Pendorong : NDBI tahun 2015



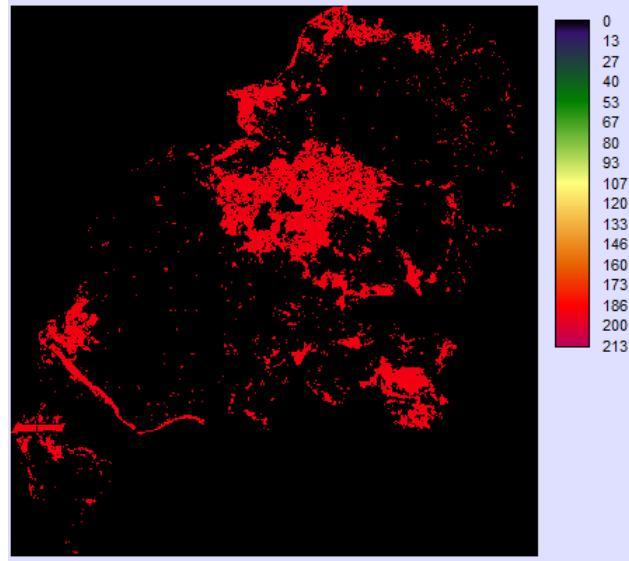
Faktor pendorong : NDVI tahun 2010



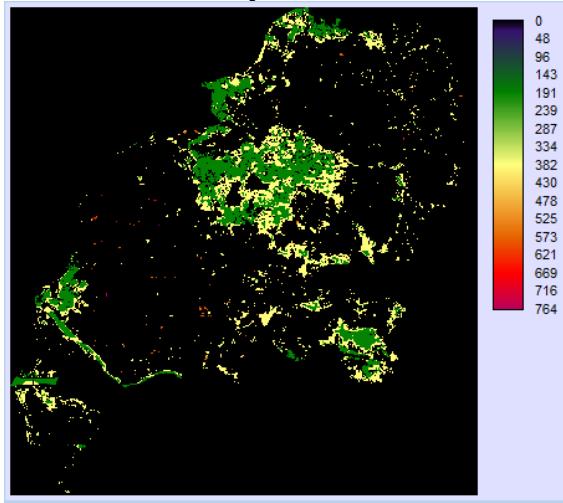
Faktor Pembatas : NDWI tahun 2010



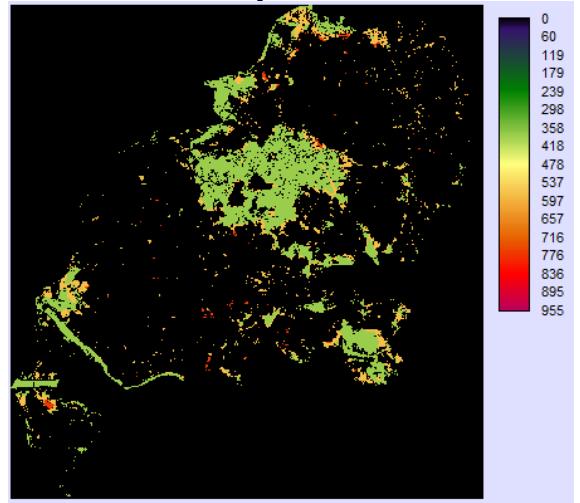
Hasil MCE



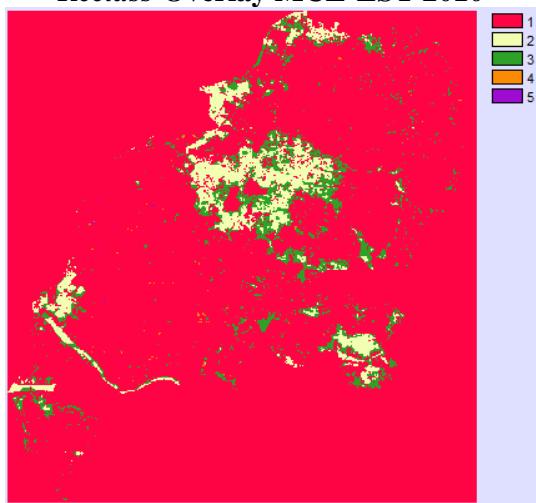
Hasil Overlay MCE- LST 2010



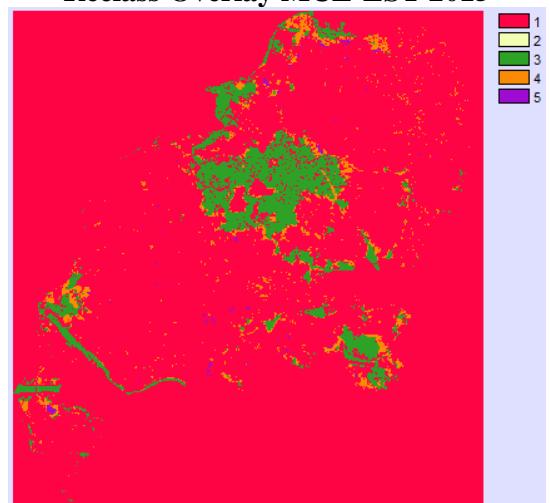
Hasil Overlay MCE-LST 2015



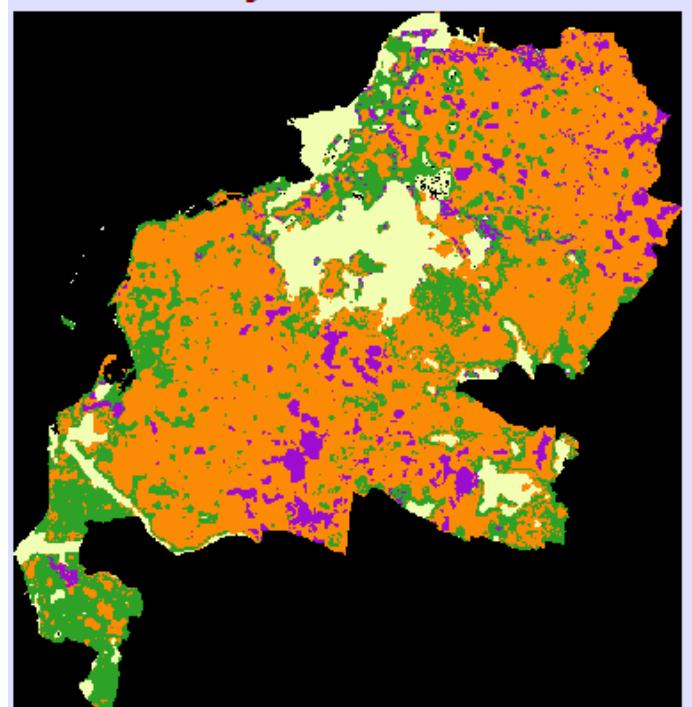
Reclass Overlay MCE-LST 2010



Reclass Overlay MCE-LST 2015

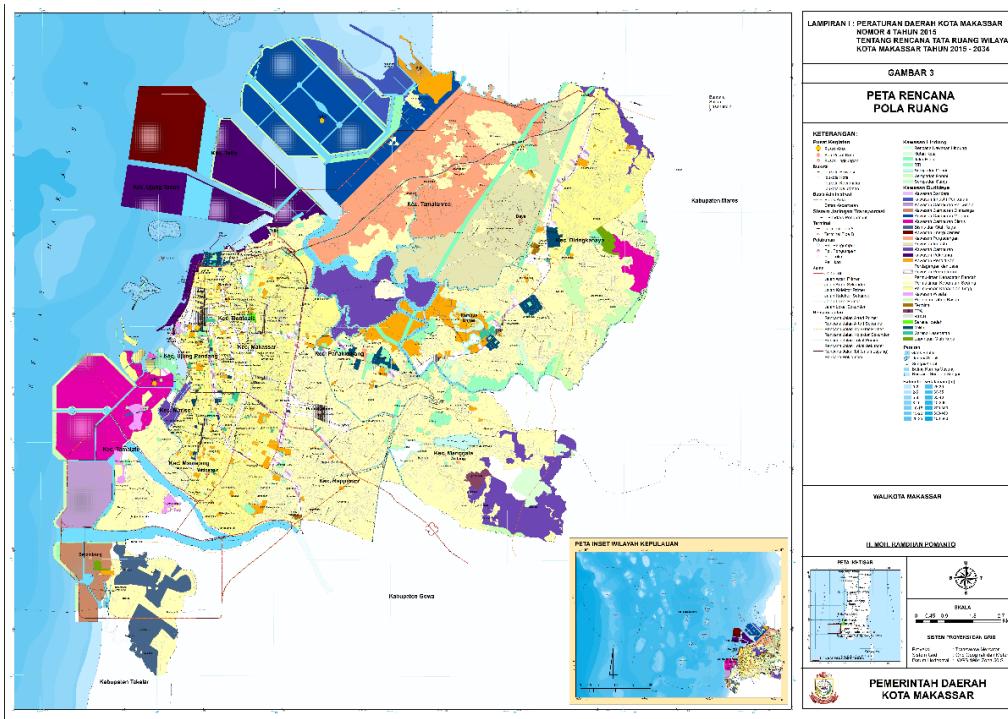


Project LST 2025

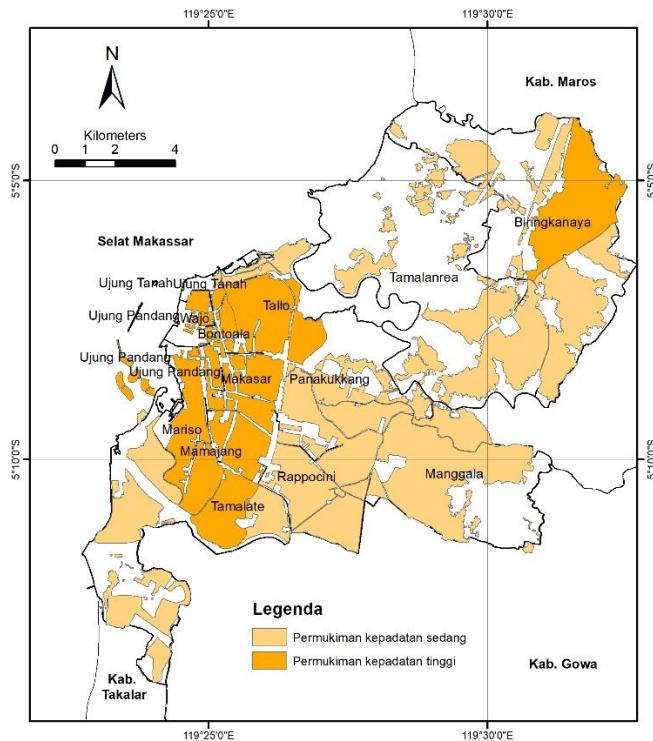


LAMPIRAN 3 : Pengolahan Data SUHII

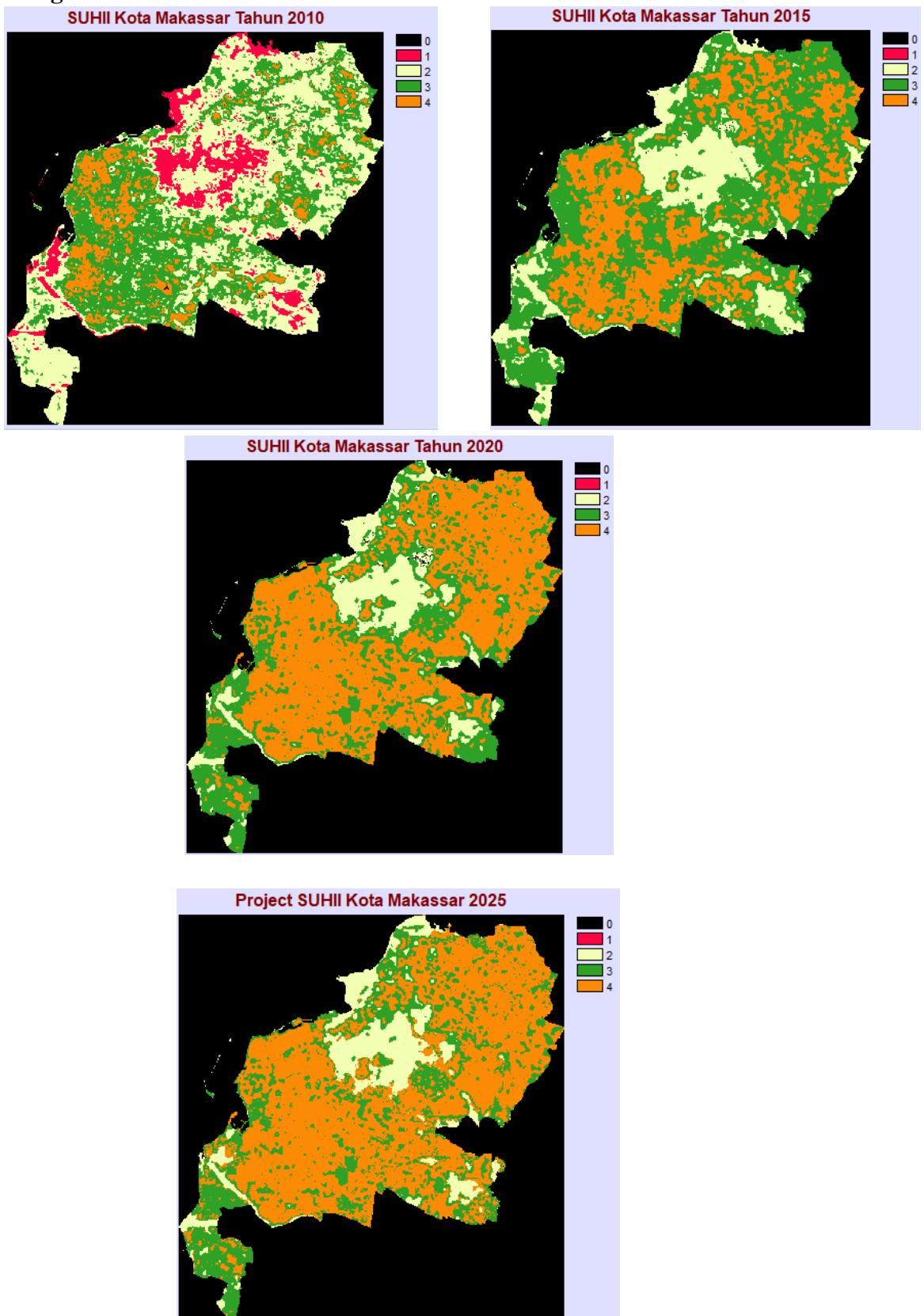
1. Peta Rencana Pola Ruang Kota Makassar tahun 2015-2034 dari Bappeda Kota Makassar



2. Hasil digitasi alokasi rencana permukiman Kota Makassar tahun 2015-2034



3. Pengolahan Model Prediksi SUHII tahun 2025



PETA PREDIKSI SUHII KOTA MAKASSAR TAHUN 2025

