THESIS

QUANTIFYING FLOODPLAIN HEALTH IN THE CONTIGUOUS UNITED STATES USING AN INDEX OF INTEGRITY

Submitted by

Kira Simonson

Department of Civil and Environmental Engineering

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Summer 2022

Master's Committee:

Advisor: Ryan Morrison

Peter Nelson
Ellen Wohl

ABSTRACT

QUANTIFYING FLOODPLAIN HEALTH IN THE CONTIGUOUS UNITED STATES USING AN INDEX OF INTEGRITY

Despite the numerous hydrological, geological, and ecological benefits produced by floodplain landscapes, floodplains continue to be degraded by human activities at a much higher rate than other landscape types. Although this large-scale landscape modification has been widely observed, a comprehensive, national dataset quantifying the degree to which human activities are responsible for this degradation has not previously been evaluated. Floodplain integrity can be defined as the ability of a floodplain to support essential environmental functions that sustain diversity and ecosystem services through geomorphic, hydrologic, and ecological dynamics. In this research, I seek to analyze floodplain integrity at a national scale for the United States by spatially quantifying the impact of anthropogenic stressors on essential floodplain functions. I assess the prevalence of human modifications through widely available geospatial datasets, which I then use to quantify indicators of floodplain health for five essential floodplain functions. The five essential floodplain functions include flood attenuation, groundwater storage, habitat provision, sediment regulation, and organics and solute regulation. Rather than focusing solely on the ecological health within the floodplain, I develop a more comprehensive integrity evaluation by assessing both the biological and hydrogeomorphic functioning ability of the floodplain. I extend a previously established methodology for quantifying floodplain integrity to better understand the impact that human development has had on floodplain health and critical floodplain functions at the national scale. Additionally, I apply this methodology using land use

change data for a 60-year period to analyze how land use has impacted floodplain integrity over time. Quantifying the health of spatially explicit floodplain elements will allow for restoration efforts to be targeted to the areas in most desperate need of preservation.

ACKNOWLEDGMENTS

I would not have been able to reach where I am today without immense encouragement and support from my community. It's easy to forget what an incredible opportunity it is to pursue higher education, and I want to thank those who have allowed me to do so at Colorado State.

Thank you to my advisor, Dr. Ryan Morrison, for being a phenomenal research advisor these past few years. Your mentorship has truly fostered so much growth in me as a student and as a young adult navigating post-undergraduate life during a global pandemic. To my committee members, Dr. Peter Nelson and Dr. Ellen Wohl, thank you for your knowledge and instruction in and out of the classroom. Spending time in the river during each of your courses is one of the highlights of my graduate experience.

To my family, friends, and classmates- thank you for believing in me. I couldn't do it without you.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 Introduction

1.1 Importance of floodplains

Floodplains are vitally important ecosystems, having considerable biological productivity and incredibly diverse ecological communities (Tockner and Stanford, 2002). Floodplains are responsible for numerous environmental benefits including flood reduction (Burt, 1997; Opperman et al., 2009; Wohl, 2021), groundwater storage (Acharya, 2000; Boulton et al., 2010; Brunke and Gonser, 1997), sediment regulation (Fryirs, 2013; Wohl, 2021), organics and solutes regulation (Hopkins et al., 2018; Olde Venterink et al., 2006), and habitat provisioning (Amoros and Bornette, 2002; Craig et al., 2008; Tockner et al., 1999). Floodplains also provide important ecosystem services, such as supplying groundwater for consumption and agriculture, ensuring productive soils for farming, supporting fisheries, providing recreational opportunities, and holding cultural importance (Fischer et al., 2019; Opperman et al., 2017; Tomscha et al., 2017).

However, floodplains are amongst some of the most endangered ecosystems, disappearing at a rate much higher than other landscape types (Tockner and Stanford, 2002). The key environmental functions that floodplains provide are often seen as incompatible with human development (Opperman et al., 2017). For example, significant inundation events are typically equated with disaster in industrial civilizations, and landscape-altering measures are often taken to prevent such flooding (Johnson et al., 2020). These measures to minimize flooding include the construction of dams and levees and the straightening and dredging of channels (Knox et al., 2022; Opperman et al., 2017). Additionally, the use of floodplains for navigation and hydropower has severely altered the original landscape and led to impaired connectivity between channels and floodplains (Fischer et al., 2019; Habersack et al., 2015; Nilsson et al., 2005). This anthropogenic development is typically responsible for a decreased range of ecosystem services

and environmental functions able to be provided by the floodplain (Ward et al., 1999). Globally, the elimination of floodplains through human development is responsible for the reduction of floodplains by more than 50% (Snyder et al., 2002).

The abundance of floodplains is decreasing at an alarming rate, requiring large-scale targeted restoration and conservation efforts (Opperman et al., 2017). Restoration efforts can not only improve the health of floodplains, but also can provide benefits to society through the re-establishment of ecosystem services. For instance, the restoration and conservation of floodplains can help mitigate future flooding disasters by allowing large floods to be attenuated before reaching downstream communities (Kousky and Walls, 2014; Schober et al., 2015).

Although the benefits of floodplain restoration have been shown, a robust identification and evaluation of floodplains impacted by human development has not been conducted within the contiguous United States (CONUS). To sustain the future of floodplains and ensure the longevity of their environmental health and ability to perform ecosystem services, "the only hope … lies with highly enlightened management and restoration efforts" (Tockner and Stanford, 2002). As a first step, Tockner and Stanford (2002) state that "a more robust and predictive understanding of how changes in hydrological connectivity and habitat fragmentation affect floodplain communities" will be essential to targeting restoration efforts.

1.2 Floodplain functionality

Although floodplains are an essential ecosystem, a comprehensive dataset representing floodplain functionality at a national level does not exist. Efforts have been made to assess floodplain health at a catchment level (Munné et al., 2003; Sun et al., 2017), but these types of assessments typically focus on evaluating the biological health of the riparian zone, rather than the overall functionality of a floodplain. Similar to floodplain functionality, Thornbrugh et al.

(2018) noted the lack of a holistic quantitative assessment of watershed functionality for the CONUS. One approach for such an assessment would be to compare a study watershed to an unaltered, reference watershed. However, since completely unaltered watersheds are largely non-existent (Stoddard et al., 2006), Thornbrugh et al. (2018) propose an alternate framework. To address the absence of watershed health data on a national scale, Thornbrugh et al. (2018) develop a methodology for assessing and mapping an Index of Watershed Integrity (IWI). The IWI defines watershed integrity with an adaptation of the definition of environmental integrity, where environmental integrity is "the capacity of a system (and its sub-components) to support and maintain the full range of ecosystem processes and functions essential to the long-term sustainability of its it is [sic] diversity and natural resources" (Flotemersch et al., 2016). This definition was proposed by Flotemersch et al. (2016) to act as a single, adaptable definition for environmental integrity, especially as it relates to anthropogenic influences (Karr, 1981).

Stemming from the definition proposed by Flotemersch et al. (2016), the quantitative assessment of watershed functionality evaluates the ability of a watershed to produce essential services. The concept of an index of integrity can be extended to floodplains, as seen in the Index of Floodplain Integrity (IFI) methodology produced by Karpack et al. (2020). To assess floodplain health at a national scale, I advance and expand the framework established by Karpack et al. (2020), which was only applied within the state of Colorado, USA, to the CONUS.

Building on the previously outlined definition of environmental integrity (Flotemersch et al., 2016), Karpack et al. (2020) defined floodplain integrity as "the ability of a floodplain to support essential geomorphic, hydrologic, and ecological functions that maintain biodiversity and ecosystem services provided to society." This definition focuses on the holistic integrity of a

floodplain's health and its ability to perform essential functions. The index of integrity framework allows for a comprehensive assessment of floodplain health, rather than just an assessment of the ecological status of floodplains. Defining floodplain integrity requires an explicit selection of the critical functions provided by floodplains. The most vital floodplain functions selected by Karpack et al. (2020) are; (i) flood reduction, (ii) groundwater storage, (iii) sediment regulation, (iv) organics and solute regulation, and (v) habitat provisioning. For each of the five critical functions, Karpack et al. (2020) identify the human-induced stressors that have the highest potential to reduce floodplain functioning.

i. Flood reduction: Floodplains attenuate fluxes of water and reduce flood volume by storing water (Burt, 1997; Wohl, 2021). Human development that lowers floodplain storage ability or increases flood stage impacts the capacity for flood reduction (Konrad, 2003; Sharma, 2017). Levees have a high potential to stress flood attenuation, as they cut off floodplain connectivity (Amoros and Bornette, 2002; Knox et al., 2022; Wheater and Evans, 2009). Roads and railroads are additional stressors to flood attenuation, as they can disconnect floodplains and increase and channelize runoff (Blanton and Marcus, 2009; Tarolli and Sofia, 2016). Land cover changes that reduce vegetation or increase urbanization also inhibit flood attenuation (Wheater and Evans, 2009).

ii. Groundwater storage: Floodplains are essential for regulating and recharging groundwater stores through vertical hydraulic connectivity (Acharya, 2000; Boulton et al., 2010; Brunke and Gonser, 1997). Vertical connectivity can be reduced with increasing barriers to infiltration. Such barriers include increased impermeable area, channelization of flow, and colmation (clogging in the top

layer of alluvial sediments). Groundwater storage ability can also be negatively impacted by the loss of wood and vegetation and excessive groundwater pumping (Brunke and Gonser, 1997; Wheater and Evans, 2009).

iii. Sediment regulation: Floodplains regulate sediment transport and fluxes through river systems by acting as both sediment sources and sinks (Fryirs, 2013; Wohl, 2021). The balance between sediment supply and storage in floodplains can be heavily disrupted by anthropogenic land cover change (Wohl et al., 2015). Increased agricultural area in floodplains increases erodibility and sediment supply (Knox, 2006; Wheater and Evans, 2009). The loss and removal of riparian vegetation impacts the sediment supply, as riparian vegetation can filter suspended particles and reduce sediment yield (Brunke and Gonser, 1997; Wheater and Evans, 2009). The channelization of flow caused by roads and railroads can lead to higher flow volumes and erosion, which are stressors to sediment regulation (Tarolli and Sofia, 2016). Sediment mobilization is also dependent on overbank flows, which can be altered with modifications to the timing and magnitude of peak flow events (Fryirs, 2013; Wohl et al., 2015).

iv. Organics and solute regulation: Floodplains regulate and store particulate organic matter and nutrients for watersheds by intermittent wetting cycles (Hopkins et al., 2018). The retention of both sediment and nutrients in floodplains can help to regulate water quality by reducing stream loads (Hinshaw et al., 2022; Loomis et al., 2000; Noe and Hupp, 2009; Olde Venterink et al., 2006; Pinay et al., 2002; Wohl, 2021). Organics and solute regulation in floodplains is also impacted by alterations to hydrologic flows (Craig et al., 2008; Tockner et al., 1999).

Hindered vertical connectivity to groundwater negatively impacts nutrient regulation (Brunke and Gonser, 1997; Burt, 1997). Overland flow interception (Tockner et al., 1999) and loss of vegetation (Craig et al., 2008; Junk et al., 1989; Sutfin et al., 2016) also reduce nutrient processing ability.

v. Habitat provisioning: Floodplains have high biological productivity and support a wide range of ecological communities (Craig et al., 2008; Tockner et al., 1999). Increased developed and agricultural area and loss of vegetation reduce biodiversity and intensify pollution (Harper et al., 1997; Tockner et al., 1999). Species invasion that is driven by human influence can also adversely impact floodplain habitats (Tockner et al., 1999). Alteration to overland flow connectivity (Beevers et al., 2012; Ward and Stanford, 1995) and hydrologic flow (Amoros and Bornette, 2002; Brunke and Gonser, 1997; Harper et al., 1997; Tockner et al., 1999; Ward et al., 1999) can harm floodplain ecosystems and biodiversity.

The selected essential functions allow for a complete assessment of the geomorphic, hydrologic, and ecological performance of a floodplain. Measuring the capacity for a floodplain to support critical functions will provide insight into the impact that human development has had on floodplain health.

1.3 Trajectories of floodplain integrity

The degree to which anthropogenic activity has impacted floodplains over the last century is broadly understood but not quantified. This lack of research studying the historical effects of human activity exists not only for floodplains but for the entire river ecosystem (Wohl, 2019). Wohl (2019) defines legacy effects as "any persistent change in a natural system resulting

from human activities" and emphasizes the complexity of analyzing the full scope of these legacy effects on river corridors. It is nearly impossible to evaluate the unaltered state of river corridors in the CONUS, as only 2% of American rivers remain unimpacted by human activity (Graf, 2001). Similarly, up to 90% of floodplains in Europe and North America have been cultivated (Erwin, 2008). Over time, floodplains have also been significantly altered by levees (Knox et al., 2022). Minimal data exist to measure how this extensive cultivation and alteration has altered floodplain health over time.

The interdependent relationship between landscapes in river corridors makes predicting the river system's response to human activities complex (Amoros and Bornette, 2002; Erwin, 2008; Ward, 1989). Alterations to river processes driven by human activity have been shown to modify the resilience of the system and shift the stable regime in river corridors, illustrating the difficulty associated with quantifying river health throughout time (DeBoer et al., 2022). While trajectories of floodplain health have been qualitatively described (Tockner and Stanford, 2002; Wheater and Evans, 2009), a quantitative trajectorial analysis will help to identify long-term changes to floodplain health and provide insight into the potential consequences of human modification to floodplains (Erwin, 2008). An analysis of the trajectories of floodplain health can be used to implement strategies for improving the resilience and protecting the health of floodplains in the future.

Rajib et al. (2021) present a database quantifying land use change in the Mississippi River Basin, finding that agricultural and developed land have substantially modified floodplain functionality for a discrete 60-year period (1941-2000). The use of this dataset within the IFI methodology will provide the opportunity to evaluate how changes in floodplain health have been driven by anthropogenic land alterations over time.

1.4 Research Objectives

Given the lack of CONUS-scale assessments of floodplain integrity, including changes in integrity over time, my research objects are to:

1) Evaluate the impact that human development has had on floodplain health and critical floodplains functions. I will do this by applying an established methodology for quantifying floodplain integrity to the CONUS. I will then analyze the geospatial variations and patterns in floodplain integrity that are seen across the CONUS.
2) Analyze the impact of anthropogenic activity on floodplain integrity over time. I will apply the established methodology using historical land use data to develop a temporal trajectory of floodplain integrity in response to human development.

2 Methods

2.1 Overview

A methodology for quantitatively evaluating floodplain integrity has been developed and applied to the state of Colorado, USA (Karpack et al., 2020). I used the process described in Karpack et al. (2020) and applied it across the CONUS with a few modifications (described below). An important consideration of this methodology developed by Karpack et al. (2020) is that it only considers human impacts on floodplains and does not include naturally occurring disturbances (e.g., fires, landslides) when evaluating floodplain integrity. To apply this methodology to the CONUS, I had to confirm that the selected datasets used to represent human-induced stressors were available at a national level. Of the stressor datasets used by (Karpack et al., 2020), only the density of groundwater wells was specific to Colorado. I replaced the Colorado-specific dataset with a similar dataset for the CONUS (U.S. Geological Survey, 2022). Once I confirmed the availability of anthropogenic stressor datasets at the national level, I

calculated the relative densities of each of the datasets within the floodplain. I used the densities of the stressor datasets within the floodplain to calculate index of integrity values for each of the five floodplain functions. I combined the integrity metrics for the five functions to calculate an overall index of floodplain integrity (IFI) for the United States. The IFI values range from zero to one, with zero indicating the least stressed floodplains and one indicating the floodplains most altered by anthropogenic development. A summary of the overall IFI calculation process can be seen in Figure 1.

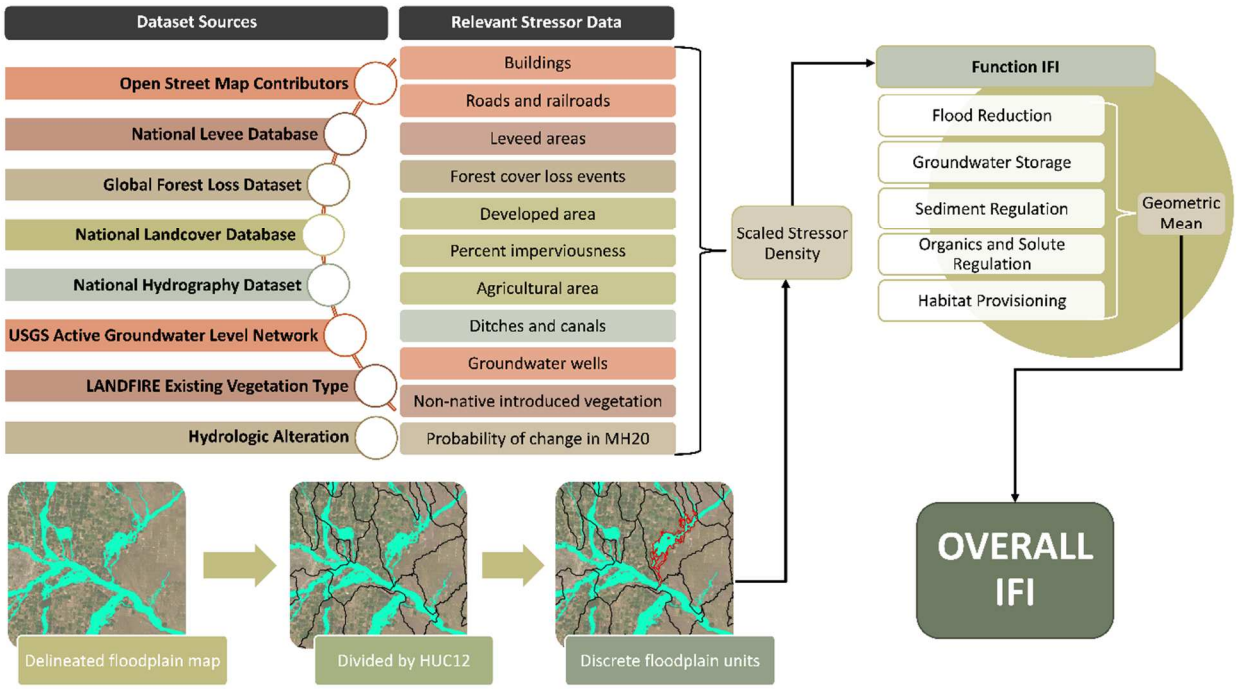


Figure 1: Overview of the IFI methodology.
Dataset sources and relevant stressor data are coordinated by matching colors.

I selected the Mississippi River Basin (MRB) for the scope of the temporal trajectory analysis because of the availability of a robust and publicly accessible historical land use dataset for the area. This dataset provided the best available opportunity to evaluate the trajectory of IFI values within the CONUS. The MRB encompasses 41% of the United States (Rajib et al., 2021) and can provide information regarding the integrity trajectory for a considerable portion of the CONUS. Although there is a lack of national historical data for many of the stressor datasets, the

MRB dataset can be utilized to quantify changes in IFI driven specifically by land use change. The dataset detects land use changes in the Mississippi River Basin floodplain for seven land use classifications. Two of these land use classifications, agricultural and developed land, directly impact the IFI calculation.

I used the historical land use data mapped in five-year increments from 1941 to 2000 for two of the critical floodplain function stressor datasets (agricultural area and developed area). All five floodplain functions are impacted by at least one of these two stressor datasets. Since all five critical floodplain functions are influenced by the agricultural and developed area datasets, it was possible to evaluate how floodplain functionality has been altered in response to land use change over time. I calculated function IFI and overall IFI values for the MRB floodplain for five-year increments over the 60-year period to see how land use change impacts floodplain health.

2.2 Floodplain delineation

To assess hydrologic alteration across floodplains in the United States, a delineated floodplain map is required. I obtained this delineation from a 30m resolution shapefile dataset of the 100-year undefended (without levees) floodplain across the United States (Wing et al., 2017). The floodplain boundaries in this shapefile were developed using a 2D hydrodynamic model and regionalized flood frequency estimates (Wing et al., 2017). I processed the floodplain shapefile by clipping it to the 2-digit hydrologic code Watershed Boundary Dataset for the CONUS (Seaber et al., 1987), removing isolated pixel groups less than 2,700 m^2 (3 pixels), and filling gaps of 2,700 m^2 or less. If the subdivision of the floodplain map resulted in an area of less than 2,700 m^2, the area was removed. This reduced the overall floodplain area from 740,967 km^2 to 721,799 km^2 (approximately a 2.5% reduction).

I further divided the floodplain map along 12-digit hydrologic unit code boundaries (HUC12). Only HUC12 areas with boundaries completely within the United States border were included in the analysis. This process created 78,304 unique "floodplain units" within the CONUS, with a total area of 662,566 km^2 and a resulting average floodplain unit area of 8.46 km^2. The process for creating individual floodplain units can be seen in Figure 2.

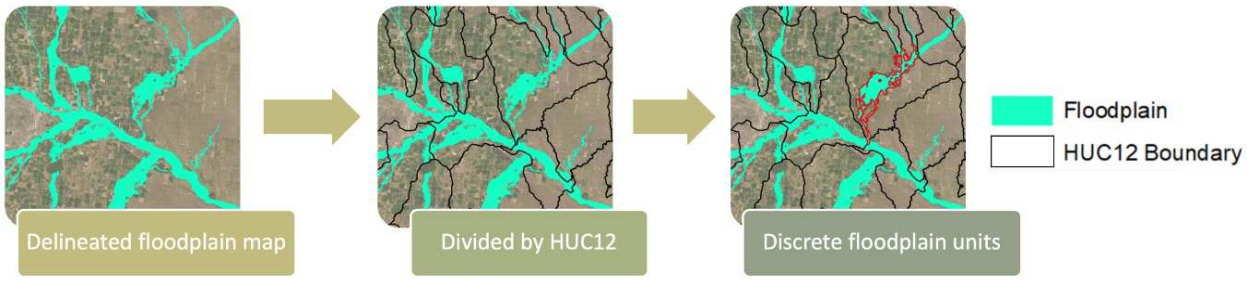


Figure 2: Visualization of process for creating individual floodplain units. Red outline represents one unique floodplain unit.

I determined the maximum stream order within each floodplain unit by using the National Hydrography Dataset Plus Version 2 (NHDplusV2) flowlines (Richard B. Moore et al., 2019). For each streamline that intersected the floodplain, a unique COMID for that streamline was also associated with a floodplain unit. Due to discrepancies between the NHD flowlines dataset and the floodplain delineation developed by Wing et al. (2017), 2,855 floodplain units do not intersect with any flowlines. This means that for these units there is not an associated maximum stream order.

For the trajectory analysis, I clipped the CONUS floodplain map to the MRB watershed delineation published by Rajib et al. (2021). This subset of the floodplain map resulted in 34,421 floodplain units with a total area of 306,664 km^2 and an average floodplain unit area of 8.91 km^2.

2.3 Selection of anthropogenic stressor datasets

Once I generated a national floodplain delineation with discrete floodplain units, it was necessary to evaluate the datasets representing the presence of anthropogenic stressors in the floodplain. Of the datasets used by Karpack et al. (2020) for the state of Colorado, I confirmed that the resolution of the data was at least that of the floodplain map (30 m) and publicly available at a national level. Only one of the previously selected datasets, groundwater wells, was unavailable at the national level. For this missing dataset, I identified a similar dataset at the national scale. It is important to note that only some datasets are direct measurements of their associated stressor, such as the representation of loss of wood and vegetation being measured directly by the forest loss cover events dataset (Hansen M. C. et al., 2013). For other stressors, I used highly representative datasets due to the limited availability of direct measurement data at the national scale. An example of this is using the prevalence of groundwater wells in the floodplain as a representation of groundwater depletion.

A key stressor dataset used to numerically quantify floodplain health is a collection of data estimating the degree of hydrologic alteration for a range of metrics to the NHDPlus V1 streamlines dataset. This dataset was created by the methods outlined in (McManamay et al., In Review) and was applied to a range of indicators described in Olden & Poff (2003). This dataset was developed by measuring hydrologic alteration at USGS gage sites by comparing the recorded flow to a modeled estimation of unaltered flow (Falcone, 2017). The measured hydrologic alteration data was extrapolated from gaged sites to ungaged streams using a random forest model across the entire US. Extrapolating the data, rather than using proxy estimations of altered streamflow, provides a much-improved method for estimating hydrologic alteration at both gaged and ungaged sites. The hydrologic alteration dataset was created along NHDPlus

Version 1 streamlines for 52 hydrologic metrics. However, an updated streamline map package, the NHDPlus Version 2 dataset, has since been published. I updated the hydrologic alteration dataset to correspond to Version 2 by "cross-walking" the data from Version 1 to Version 2 by using COMIDs of the streamlines (Richard B Moore et al., 2019). Once the hydrologic alteration dataset was updated to Version 2, I associated the hydrologic alteration metrics for each maximum order streamline with the floodplain dataset. To account for hydrologic alteration within the index of integrity calculation I used the hydrologic alteration metric "alteration to mean annual maximum flows divided by catchment area" (MH20) as a representation of the peak flow conditions most likely to activate floodplains. For a more detailed discussion of each floodplain function and its associated stressors, refer to Section 1.2.

2.4 Calculation of stressor dataset densities

After the identification and evaluation of the representative datasets (Table 1), the density of each of these stressors within the floodplain unit had to be calculated. For point, polyline, and polygon datasets, the number (count/km^2), length (km/km^2), and area (km^2/km^2) of each stressor dataset within the floodplain unit were calculated. For raster data, the process for calculating density was dataset specific. I computed agricultural area as the percentage of cells in the floodplain reported as pasture/hay or cultivated crops (NLCD classes 81 and 82). I computed developed area as the percentage of cells in the floodplain reported as low, medium, and high intensity development (NLCD classes 22, 23, and 24). I calculated forest cover loss events as the percentage of cells in the floodplain that reported forest loss events between 2000 and 2020. Percent imperviousness was computed by averaging the percent imperviousness values reported for each 30 m cell for all cells in the floodplain unit. I quantified the prevalence of invasive species by computing the percentage of cells in the floodplain reported as non-native, introduced

vegetation (LANDFIRE Existing Vegetation Type groups 701-709, 711, and 731). I averaged the hydrologic alteration MH20 values for all the maximum stream order segments within each floodplain to aggregate the hydrologic alteration metrics for the floodplain units. All stressor densities within the floodplain were calculated using R programming (RStudio Team, 2015) scripts (Appendix B).

Table 1: Floodplain function stressors and corresponding datasets.

Function	**Stressor**	**Dataset**	**Attributes**
Flood reduction	Reduced storage volume	Buildings[1]	Polygon, accessed Jan 20th, 2022
	Floodplain disconnection	Leveed areas[2]	Polygon, accessed Jan 29th, 2021
	Overland flow interception	Roads and railroads[1]	Polyline, accessed Jan 20th, 2022
	Land cover change	Forest cover loss events[3]	Raster, 30m, loss events from 2000-2020
	Land cover change	Developed area[4]	Raster, 30m, 2019 version
Groundwater storage	Impermeable surface	Percent imperviousness[4]	Raster, 30m, 2019 version
	Channelized overland flow	Ditches and canals[5]	Polyline, 1:100,000 scale, NHD+ V2
	Colmation	Agricultural area[4]	Raster, 30m, 2019 version
	Loss of wood and vegetation	Forest cover loss events[3]	Raster, 30m, loss events from 2000-2020
	Lowered water table	Groundwater wells[6]	Points, accessed Jan 20th, 2022
Sediment regulation	Land cover change	Agricultural area[4]	Raster, 30m, 2019 version
	Loss of wood and vegetation	Forest cover loss events[3]	Raster, 30m, loss events from 2000-2020
	Overland flow interception	Roads and railroads[1]	Polyline, accessed Jan 20th, 2022
	Hydrologic alteration	Probability of change in MH20[7]	Data for NHD+ V2 polylines
Organics and solute regulation	Hydrologic alteration	Probability of change in MH20[7]	Data for NHD+ V2 polylines
	Vertical connectivity	Percent imperviousness[4]	Raster, 30m, 2019 version
	Overland flow interception	Roads and railroads[1]	Polyline, accessed Jan 20th, 2022
	Loss of wood and vegetation	Forest cover loss events[3]	Raster, 30m, loss events from 2000-2020
Habitat provision	Land cover change	Agricultural area[4]	Raster, 30m, 2019 version
	Land cover change	Developed area[4]	Raster, 30m
	Loss of wood and vegetation	Forest cover loss events[3]	Raster, 30m, 2019 version
	Species invasion	Non-native introduced vegetation[8]	Raster, 30m, 2016 version
	Overland flow interception	Roads and railroads[1]	Polyline, accessed Jan 20th, 2022
	Hydrologic alteration	Probability of change in MH20[7]	Data for NHD+ V2 polylines

1. *Open Street Map Contributors, 2022*
2. *National Levee Database; U.S. Army Corps of Engineers, 2019*
3. *Global Forest Loss Dataset; Hansen et al., 2013*
4. *National Landcover Database; Dewitz, 2021*
5. *National Hydrography Dataset Plus, Version 2; Moore, McKay, et al., 2019*
6. *USGS Groundwater Watch; U.S. Geological Survey, 2022*
7. *Hydrologic Alteration Data; (McManamay et al., In Review)*
8. *LANDFIRE Existing Vegetation Type; Rollins, 2009*

The Mississippi River Basin dataset (Rajib et al., 2021) provides raster data of seven land use classes for each year from 1941 to 2000. These seven generic land use classes were reclassified from 17 original classes to simplify the dataset. Two of these seven reclassified land use types were used in the temporal analysis of the MRB (refer to Appendix A for additional information about reclassified land types in the MRB). I calculated the percentage of cells defined as "agriculture" or "developed area" in the floodplain in five-year increments beginning in 1941 and ending in 2000 (1941, 1945, 1950, etc.). Within the IFI calculation, I used the land use data provided by the MRB dataset in place of NLCD classes for pasture/hay or cultivated crops (i.e., agricultural area) and NLCD classes for low, medium, and high intensity development (i.e., developed area), respectively. For every other stressor dataset, I used the most recently published data at the national scale to calculate the IFI values over time. Although this contemporary data does not represent the historical state of the other stressors during the 60-year study period, it allows for the analysis of change in floodplain integrity over time to be isolated specifically to land use alterations. I used a combination of both historical and recent datasets to maintain consistency with the established IFI methodology and because there is a lack of historical data for many of the anthropogenic stressor datasets on the national scale.

Once I calculated the prevalence of stressors in the floodplain by the methods outlined in Section 2.4, the density values needed to be rescaled to comparable metrics across each dataset. Although the density values reported as percentages (km^2/km^2) have a potential maximum of one, the datasets measured by count and length have no theoretical maximum value. To address this issue, all stressor datasets were rescaled from zero to one, with a value of zero indicating the absence of the stressor in the floodplain, and one representing the 90^{th} percentile of the stressor amount in the CONUS. For the three datasets that had no stressor present at the 90^{th} percentile

(canals and ditches, leveed area, and groundwater wells), the density values were scaled to the maximum observed value in the US. This stressor rescaling was done to create a consistent scale of comparison for stressor prevalence amongst all types of stressor datasets (Karpack et al., 2020). The IFI methodology serves to compare floodplain integrity values between all floodplain units present in the scope of the study. Rescaling the stressor datasets relative to either the 90th percentile or the highest maximum observed value allows for a consistent comparison of floodplain health relative to other floodplains across the CONUS.

2.5 Calculation of function specific IFI

I calculated functional IFI values for each of the five critical floodplain functions based on the rescaled stressor densities. Before computing the IFI value, I performed a Pearson Correlation Analysis between each stressor dataset to avoid overweighing any individual data source (see Appendix A). For any function IFI calculation that included two datasets with a correlation of over 0.7, only one dataset was included in the computation.

I computed the function IFI value using the following equation:

$$IFI_{i,k} = 1 - \sum_{j=1}^{n_{j,k}} \frac{S_{i,j}}{n_{j,k}}$$

Where:

$IFI_{i,k}$ is the integrity value of the i^{th} floodplain unit for the k^{th} function

$S_{i,j}$ is the scaled stressor value in the i^{th} floodplain unit for the j^{th} stressor

$n_{j,k}$ is the number of stressor datasets, j, that impact the k^{th} function

The results of the function specific IFI computation produce a negative linear relationship between stressor density and function floodplain integrity. This method assumes an equal impact of each stressor dataset on floodplain functionality.

2.6 Overall calculation of IFI

I computed the overall integrity values for each floodplain unit by the geometric mean of the function specific IFI values:

$$IFI_i = \left(\prod_{k=1}^{5} IFI_{i,k} \right)^{\frac{1}{5}}$$

Where:

IFI_i is the overall integrity value of the i^{th} floodplain unit

$IFI_{i,k}$ is the integrity value for the k^{th} function in the i^{th} floodplain unit

Computing the overall IFI value by geometric mean of the function specific IFI values reflects the importance of each individual critical floodplain function to floodplain health. By this method, a function IFI value of zero as produces an overall IFI value of zero. This emphasizes that each of the five functions is essential to overall floodplain functionality.

2.7 Comparison of IFI across the CONUS

Once I calculated the function specific and overall IFI values for the floodplain, I associated each floodplain unit with a variety of spatial attributes. I associated each floodplain unit with three geospatial categories:

1) Urban vs rural (US Census Bureau, 2019)
2) Maximum stream order (Richard B. Moore et al., 2019)
3) Ecoregion (US EPA, 2016)

I determined the intersection between each floodplain unit and the selected spatial attributes in ArcMap (ERSI, 2011). I then analyzed the geospatial distribution of these attributes and the IFI results for the CONUS.

I selected these three characteristics to analyze how IFI values vary regarding anthropogenic, hydrological, and ecological features in the CONUS. For the analysis of IFI by ecoregion, I used nine aggregated Omernik Level III ecoregions in the US (Herlihy et al., 2008; US EPA, 2016). These aggregated ecoregions were used in the National Rivers and Streams Assessments (US EPA, 2020) and were developed to minimize biological and hydrogeological differences within each region (Herlihy et al., 2008). I selected the aggregated ecoregions so that I could compare IFI values for regions with similar watershed ecology in the CONUS. Comparing IFI values by ecoregion allowed me to analyze the relationship between anthropogenic stressor prevalence, watershed ecology, and floodplain health.

Since the IFI methodology is conceptual and has no numerical meaning beyond a relative comparison between floodplain units, it is difficult to validate the methodology's performance. However, characterizing the floodplain units along the aggregated ecoregions allowed me to geospatially compare IFI values with the previously discussed IWI dataset (Thornbrugh et al., 2018). Thornbrugh et al. (2018) calculate IWI summary statistics for each of the nine aggregated ecoregions in the CONUS, which I compared to the results of the IFI methodology. I also compared the IFI dataset to two revised versions of IWI datasets (Johnson et al., 2019). The first revised dataset includes an update to the stressor scaling component of the methodology and the second revised dataset utilizes a more complex relationship between stressor density and watershed health.

3 Results

3.1 IFI in the CONUS

I calculated functional IFI and overall IFI values for 78,304 unique floodplain units within the CONUS. A statistical summary of the function and overall IFI values is shown in Table 2. A

summary histogram of the distribution of each function IFI value and overall IFI value by total floodplain area can be seen in Figure 3. The function IFI values all show a high correlation, ranging from 0.7 to 0.91 (see Appendix A). Figure 4 shows overall IFI values in the CONUS according to HUC12 boundaries. For the entire CONUS, the median IFI value is 0.762 (mean = 0.740, standard deviation = 0.152). IFI values tend to be lower in the east and coastal southeast regions of the country. Concentrations of low IFI values can also be seen within the state of California and along the outlet of the Mississippi River Basin.

Table 2: Summary statistics of IFI values for the CONUS.

	Overall IFI	**Flood reduction**	**Groundwater storage**	**Sediment regulation**	**Organics and solutes regulation**	**Habitat provision**
Minimum	0.000	0.013	0.363	0.015	0.000	0.074
Maximum	1.000	1.000	1.000	1.000	1.000	1.000
Median	0.762	0.893	0.867	0.697	0.712	0.712
Mean	0.740	0.831	0.862	0.676	0.675	0.699
Std. Dev.	0.152	0.168	0.109	0.176	0.179	0.159

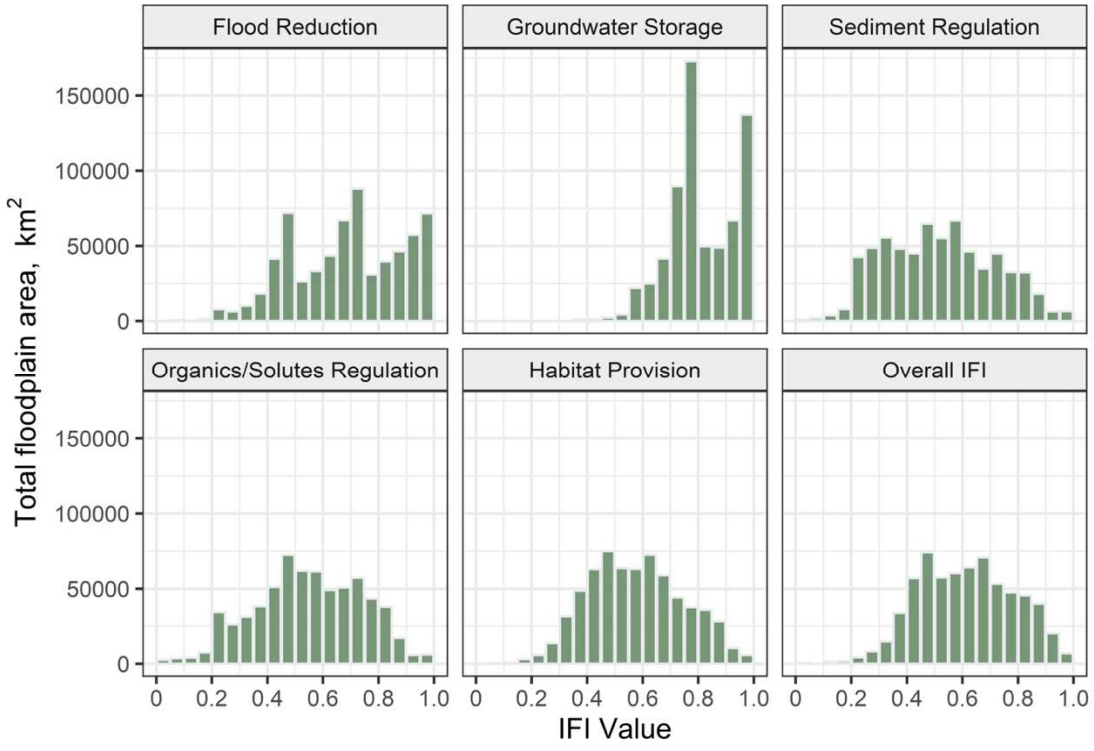


Figure 3: Histogram of IFI value distribution by total floodplain area.

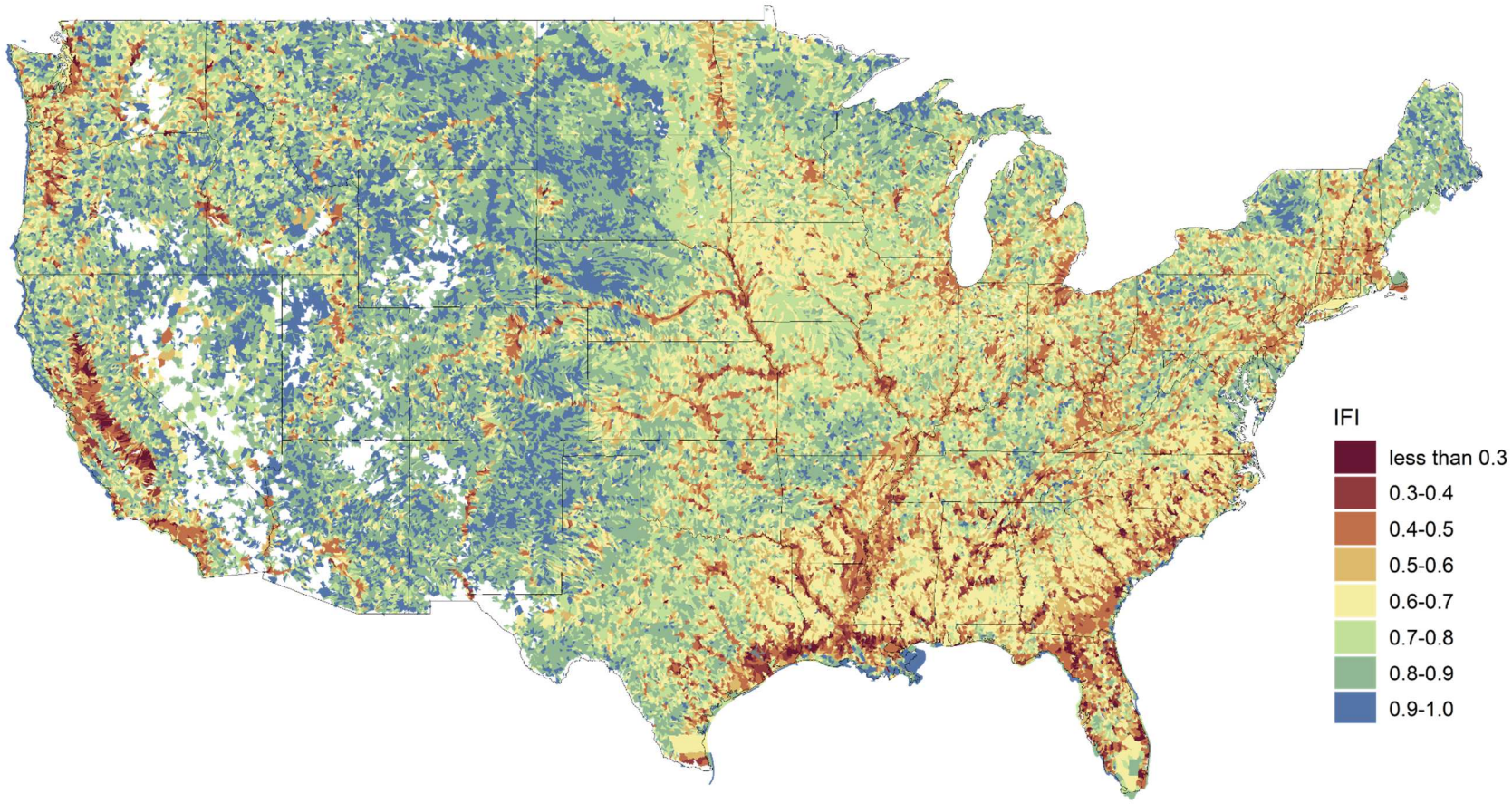


Figure 4: Overall IFI values in the CONUS.
Floodplain unit IFI values mapped to 12-digit HUC boundaries for easier visualization

An analysis of floodplains that intersected urban and rural areas (US Census Bureau, 2019) can be seen in Figure 5. The IFI values of floodplains that intersected urban areas (median = 0.557, mean = 0.561, standard deviation = 0.142) are lower than the IFI values of floodplains that did not (median = 0.790, mean = 0.769, standard deviation = 0.127). Using Tukey Honestly Significant Difference (HSD) test (Tukey, 1953), I found that the average IFI value of floodplains in urban areas is significantly different ($p<0.0001$) than the average IFI value of rural floodplains.

I also compared overall IFI values between maximum stream orders associated with each floodplain (Figure 6). Significantly different average IFI values were found between all stream orders except 1st to 2nd, 2nd to 3rd, and 4th to 5th stream order ranges. In general, IFI values tend to decrease with increasing stream order. It is worth noting that for the orders which do not exhibit a decrease in average IFI value as stream order increases, the sample size is small relative to those orders that follow the trend of decreasing IFI with increasing stream order.

I compared IFI values between nine aggregated Omernik Level III ecoregions in the US (US EPA, 2016). The highest IFI values occurred in the Northern Plains ecoregion (median = 0.878, mean = 0.855, standard deviation = 0.079), while the lowest IFI values occurred in the Coastal Plains ecoregion (median = 0.625, mean = 0.607, standard deviation = 0.142). A boxplot of ecoregions ordered by decreasing average IFI value shows that the difference between average overall IFI in each ecoregion is statistically significant for all ecoregions except for the Upper Midwest and the Southern Plains (Figure 7).

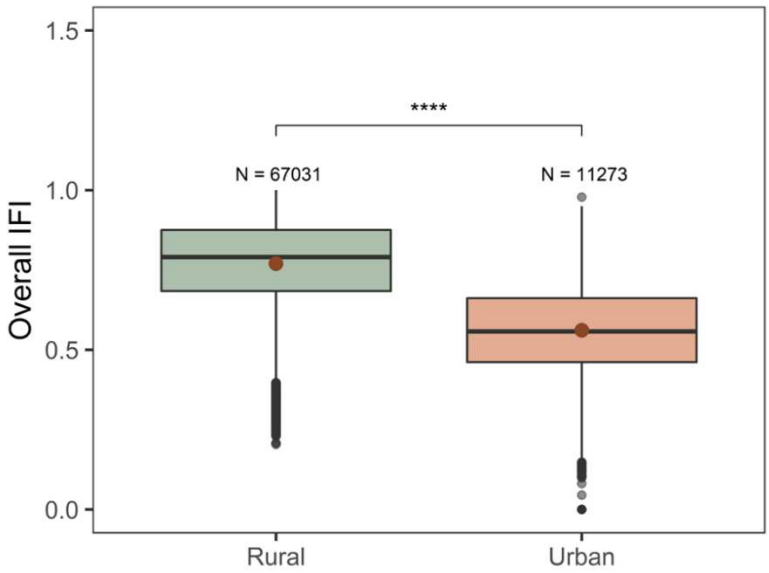


Figure 5: Analysis of IFI by urban and rural area.
*Mean IFI value is indicated by the circular points on the plot. Statistical significance between means is indicated by ns (not significant), *, **, ****, or ****, indicating the p value is >0.05, <0.05, <0.01, <0.001, or <0.0001, respectively.*

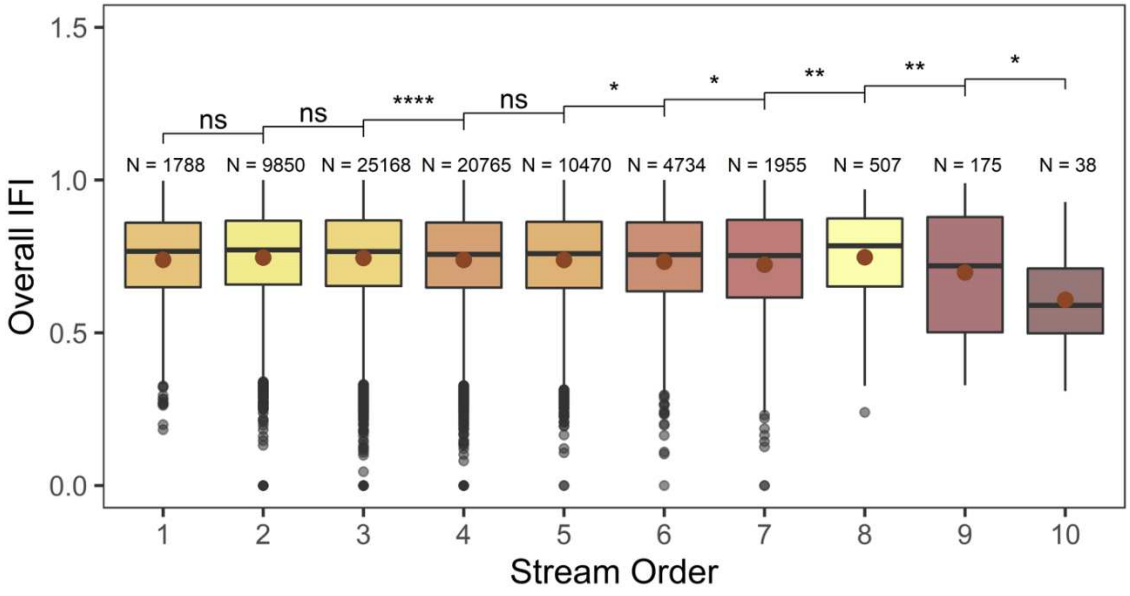


Figure 6: Analysis of IFI by maximum stream order.
*Mean IFI value is indicated by the circular points on the plot. The plot is filled with a color gradient where the darkest colors indicate lowest average IFI value and lightest colors indicate highest average IFI value. Statistical significance between means is indicated by ns (not significant), *, **, ****, or ****, indicating the p value is >0.05, <0.05, <0.01, <0.001, or <0.0001, respectively.*

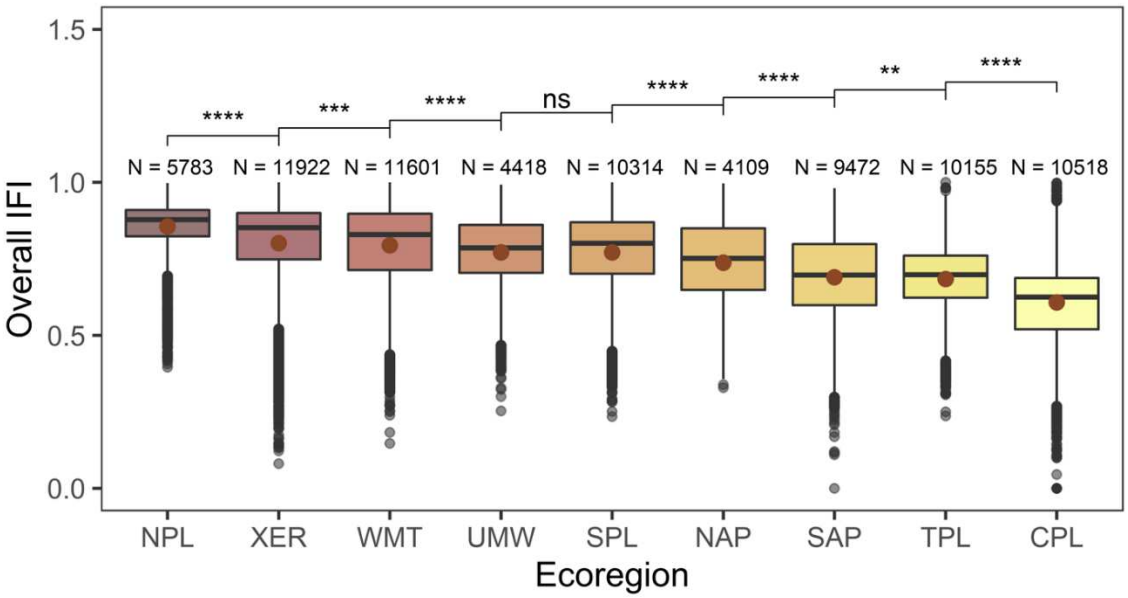


Figure 7: Analysis of IFI by aggregated Level III Ecoregions.
*Mean IFI value is indicated by the circular points on the plot. The plot is filled with a color gradient where the darkest colors indicate lowest average IFI value and lightest colors indicate highest average IFI value. Statistical significance between means is indicated by ns (not significant), *, **, ****, or ****, indicating the p value is >0.05, <0.05, <0.01, <0.001, or <0.0001, respectively. (Key: NPL = Northern Plains, XER = Xeric, WMT = Western Mountains, UMW = Upper Midwest, SPL = Southern Plains, NAP = Northern Appalachians, SAP = Southern Appalachians, TPL = Temperate Plains, CPL = Coastal Plain)*

The ratio of function IFI values to overall IFI values was analyzed to evaluate the relationship between each function and overall IFI value within the floodplain (Figure 8). Flood reduction and groundwater storage function IFI values tended to increase overall IFI, while sediment regulation, organics and solute regulation, and habitat provision function IFI values all tended to lower overall IFI.

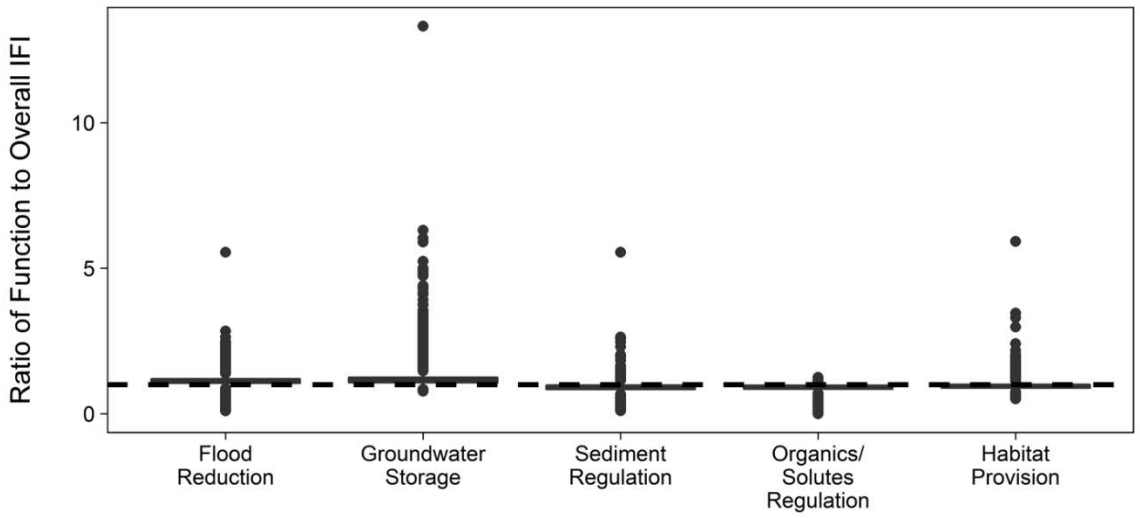


Figure 8: Ratio of functional IFI to overall IFI.
Ratios greater than one indicate that function IFI value increases the overall IFI value, while ratios less than indicate that the function value decreases the overall IFI value.

Furthermore, I compared IFI values to an original and revised IWI dataset mapped to NHDplusV2 streams for the CONUS (Johnson et al., 2019). Johnson et al. (2019) propose modifications to the original IWI dataset for the CONUS, referred to as version 1 (Thornbrugh et al., 2018) and developed two revised IWI datasets. The first revised dataset, version 2, includes a revision to the rescaling component of the IWI calculation. The second revised dataset, version 2.1, modifies the IWI methodology to include stressor-watershed function relationships and weights through random forest models (rather than assuming a negative linear relationship between stressors and watersheds). I mapped IWI values for all three versions to the maximum order NHDplusV2 stream segments intersecting each floodplain unit and then averaged within each unit. A comparison between overall IFI and each of the IWI versions for the CONUS yielded no meaningful relationships, with the highest R^2 value of 0.15 occurring between IFI and IWI version 1 (Figure 9).

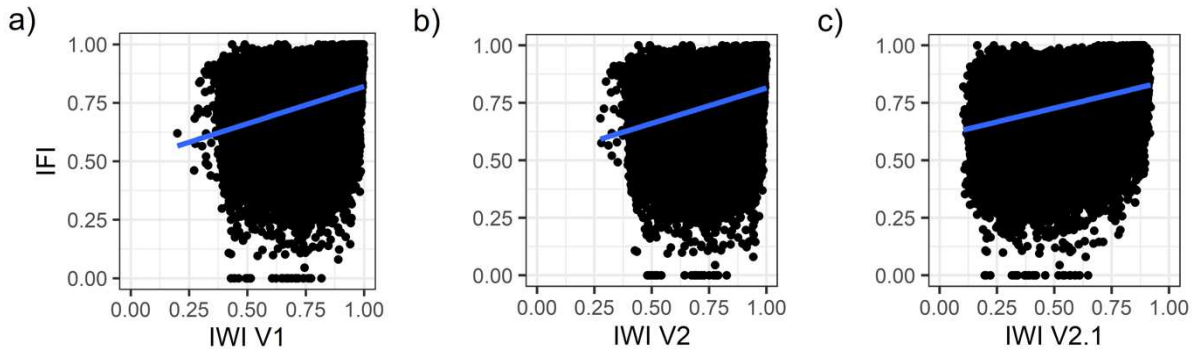


Figure 9: Comparison of Index of Floodplain Integrity values to Index of Watershed Integrity values for the CONUS.
Plots shown for a) version 1, b) version 2, and c) version 2.1.

3.2 Selected subregion analysis

Since the IFI methodology was applied at such a large scale, it is difficult to examine individual IFI values when viewing the entirety of the dataset. To more closely understand how IFI values vary on a regional scale, I analyzed function and overall IFI values for three selected subregions within the CONUS delineated along 4-digit hydrologic unit codes (Figure 10).

The San Joaquin subregion is in California with a total floodplain area of 8,248 km^2 and 435 individual floodplain units. The average overall IFI value in the San Joaquin subregion is 0.691. The minimum average function IFI value for the region is organics and solute regulation (mean = 0.627). Spatial variability in average IFI values in the subregion can be seen in Figure 10a, with a high concentration of low IFI values in the western section of the subregion contrasting against the moderate to high IFI values of the eastern section.

The Missouri-White subregion spans Nebraska and South Dakota. The total floodplain area within the subregion is 1,740 km^2 with 594 individual floodplain units. The average overall IFI value within the region is 0.860. The minimum average function IFI value for the region is sediment regulation (mean = 0.791).

The Lower Mississippi-Yazoo subregion is in the lower Mississippi River Basin and spans Arkansas, Louisiana, Mississippi, and Tennessee. The total floodplain area within the subregion is 17,969 km^2 with 391 individual floodplain units. The average overall IFI value within the region is 0.561. The minimum average function IFI value for the region is sediment regulation (mean = 0.451).

A summary of the average function and overall IFI results for the selected regions is shown in Table 3. In addition, Figure 11 shows the distribution of IFI values for each functional and overall integrity values (for more detailed summary statistics of each of the selected regions see Appendix A).

Table 3: Summary of IFI values for selected subregions.

Subregion	**Overall IFI**	**Flood reduction**	**Groundwater storage**	**Sediment regulation**	**Organics and solutes regulation**	**Habitat provision**
San Joaquin	0.691	0.782	0.832	0.629	0.627	0.644
Missouri-White	0.860	0.968	0.944	0.791	0.808	0.813
Lower Mississippi-Yazoo	0.561	0.616	0.772	0.451	0.539	0.517

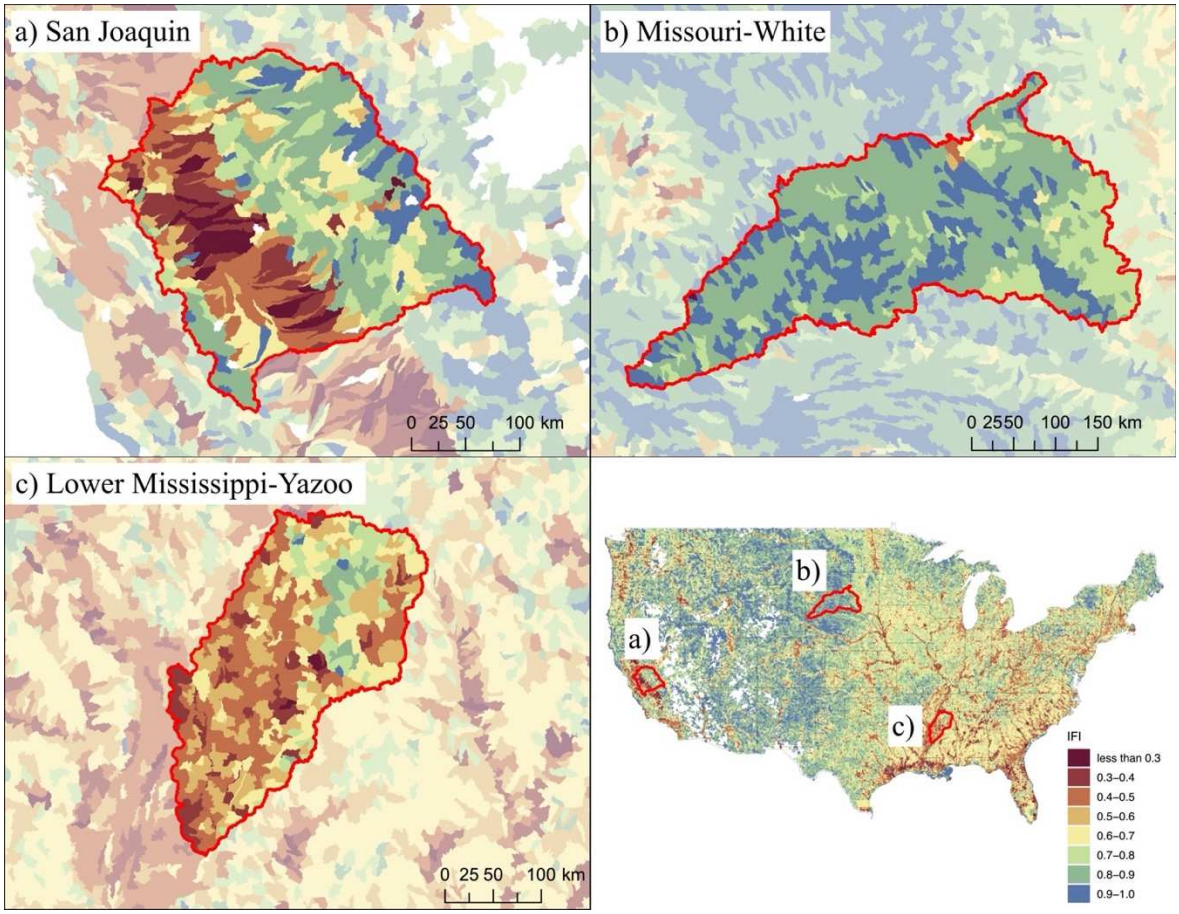


Figure 10: IFI results for the a) San Joaquin, b) Missouri-White, and c) Lower Mississippi-Yazoo HUC4 subregions.

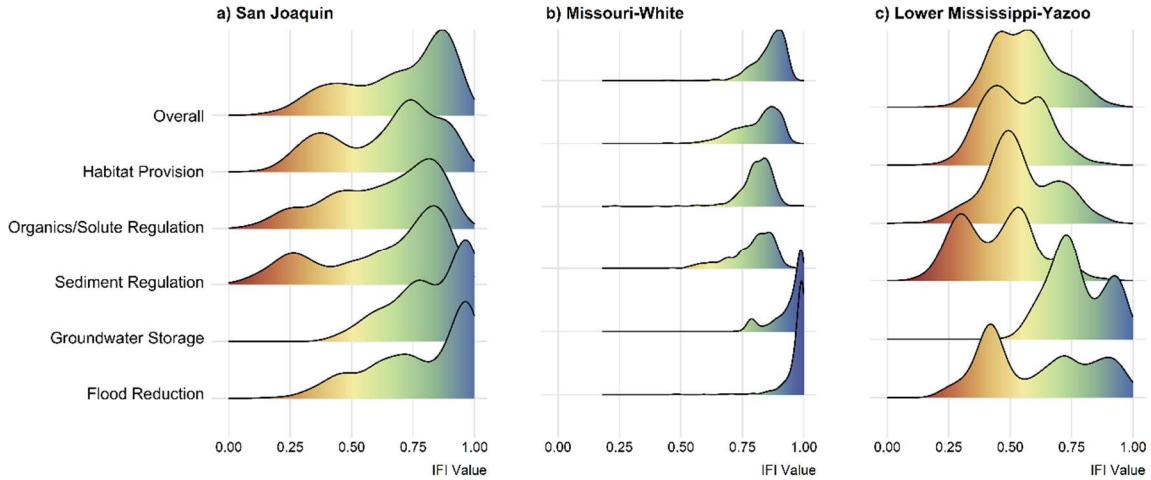


Figure 11: Density plots IFI values for the selected subregions.
Plots shown for a) San Joaquin, b) Missouri-White, and c) Lower Mississippi-Yazoo HUC4 subregions.

3.3 Trajectory of IFI from 1941 to 2000

I calculated the functional and overall IFI values in five-year increments within the Mississippi River Basin watershed starting in the year 1941 and ending in 2000, and I compared overall IFI values between each time step (Figure 12). The overall IFI values were compared between each consecutive time step using Tukey Honestly Significant Difference (HSD) test (Tukey, 1953). No significant difference was found between each time step for overall or individual function IFI values from year to year. To better visualize the difference in overall IFI values over the time period, I plotted a map of the directionality of change in overall IFI values (Figure 13). Both positive and negative changes in overall IFI values can be seen, although this change is minimal. The density curve of the percent change in IFI from 1941 to 2000 also reflects the minimal change seen in IFI values within the MRB (Figure 13).

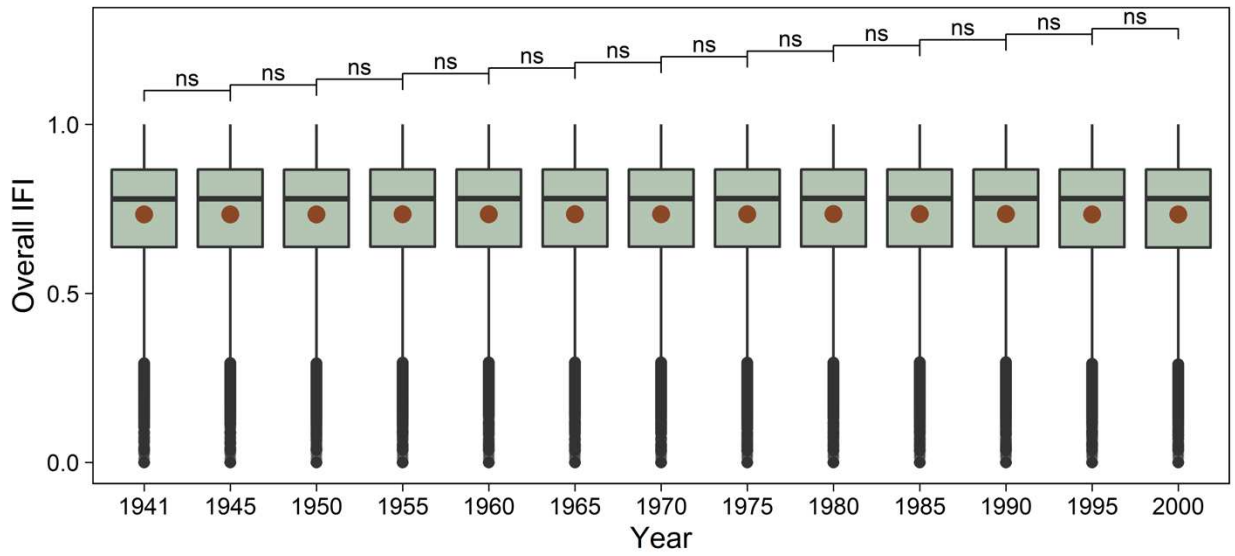


Figure 12: Analysis of IFI within the MRB by year.

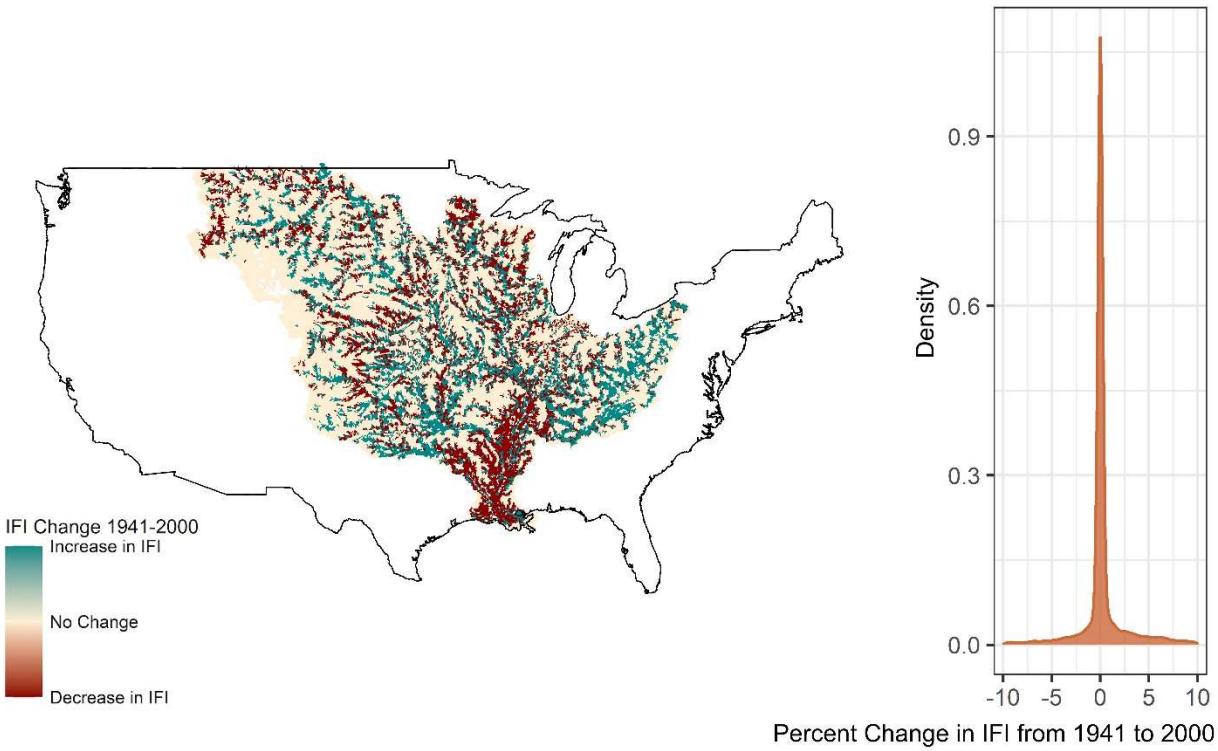


Figure 13: Directionality of change and density curve of percent change for overall IFI in the MRB from 1941 to 2000.

4 Discussion

Applying the IFI methodology at the national scale allows for a quantitative measure of individual floodplain health relative to other floodplains across the country. The geospatial results of the integrity index provide a visual indication of the degree to which anthropogenic activity has impeded floodplain functionality. Since the IFI methodology provides quantitative information on floodplain health compared to other floodplains within the scope of study, those areas most in need of restorative efforts nationally can be identified. Additionally, the flexibility of the IFI methodology means that it can be repeated at smaller scales to gain a better understanding of the variability in floodplain functionality at a more localized level.

4.1 IFI in the CONUS

The observed spatial heterogeneity in IFI across the CONUS (Figure 4) suggests that the IFI methodology is successful in identifying the floodplains most impacted by human

development. It is not surprising to see that IFI values tend to be lower in higher order streams (Figure 6), as this is historically where human development and anthropogenic activity are concentrated (Böck et al., 2018). This negative relationship between human development and floodplain health (Bunn and Arthington, 2002; Hupp et al., 2009; Peipoch et al., 2015; Smucker and Detenbeck, 2014) is also seen when comparing floodplains in rural areas to floodplains in urban areas (Figure 5). The lower average IFI values observed in urban areas confirm that floodplain health is worsened by human modifications.

4.1.1 Comparison between selected regions

I selected three subregions within the CONUS to more closely analyze the national application of the IFI methodology (Figure 10). In addition to analyzing the IFI values for each subregion, I investigated the variability in stressor dataset densities for each area. A detailed plot of the scaled stressor densities within each region can be found in Appendix A.

I chose the San Joaquin subregion (Figure 10a) to investigate the area in California with a high concentration of lower IFI values that I observed in Figure 4. The lower IFI values are seen in the floodplain in the western portion of the subregion near a densely populated part of California (Wade and Ebert, 2016). The intense agricultural development in the Salinas Valley (Farnsworth and Milliman, 2003) is reflected in the spatial distribution of IFI values seen in Figure 10a. Both high and low IFI values can be seen in the floodplain units of the subregion (Figure 11a), showing how strongly human development is impacting floodplain health.

I selected the Missouri-White subregion (Figure 10b) to analyze the distribution of IFI values in a relatively unaltered area in the western portion of the country. Most of the land cover in this subregion is classified as NLCD Grassland/Herbaceous (Dewitz, 2021). This lack of anthropogenic alteration to the subregion is reflected in the high IFI values observed in the region (Figure 11b).

I analyzed the Lower Mississippi-Yazoo subregion (Figure 10c) to see how IFI values respond in floodplains with a high percentage of leveed area, which is common along the lower Mississippi (Remo et al., 2009). The Lower Mississippi Basin has also experienced intense deforestation for intensive agriculture, severely limiting the area's ability to sustain ecosystem functioning (Hanberry et al., 2015). I wanted to analyze a subregion in the southeastern portion of the country where lower IFI values tend to be seen (Figure 4). Within the Lower Mississippi-Yazoo subregion, more than 68% of the floodplain intersects with levees (U.S. Army Corps of Engineers, 2019). The high density of levees in the lower Mississippi basin has been shown to impact the sediment transportation (Kesel, 2003; Mossa, 1996) and the flood reduction (Remo et al., 2018) capacity of the system. The prevalence of this stressor is reflected in the spike of lower flood reduction and sediment regulation function IFI values seen in Figure 11c.

The spatial variability in IFI values in this subregion can be seen in the density plot as well. All curves for both function and overall IFI within the subregion display a bi-modal distribution, reflecting how both high and low IFI values are present. However, the higher density of lower IFI function values tends to bring the overall average IFI in the region down.

4.1.2 Comparability of IFI in the CONUS

It is difficult to validate the IFI methodology since it produces a comparative dataset with values that have no physical meaning beyond an index of functionality relative to other floodplains in the scope of study. In an attempt to validate the IFI methodology applied at the national level, I compared the IFI results to three versions of the index of watershed integrity methodology (Johnson et al., 2019; Thornbrugh et al., 2018). I was surprised that there was not a higher correlation between IFI values and the three versions of IWI values in the CONUS. The IWI datasets focus on primarily hydrologic functions within a river and watershed, and the IWI methodology includes components such as water chemistry, which may be an explanation for

why such little correlation is seen. Reduced connectivity between active channels and their floodplains means that river health may not always be a good indicator of overall watershed and floodplain functionality (Fuller and Death, 2018; Stone et al., 2017).

Although there was little correlation between the IWI and IFI values across the nation, I observed similar spatial patterns of IFI and the version 1 IWI values in the nine aggregated Omernik Level III. The three regions with the highest IFI and IWI values are consistent when aggregated by ecoregion (Northern Plains, Xeric, and Western Mountains). The two ecoregions with the lowest IFI and IWI values are also the same (Temperate Plain and Coastal Plain). This similarity in spatial distribution by ecoregion between IFI values and IWI values suggests that although watershed health may not be a good way to validate floodplain functionality, the IFI methodology is successful in identifying the spatial patterns of broader river corridor health across the CONUS.

All three of the ecoregions with the highest IFI values have high percentages of grassland/shrub land cover type (NPL = 68%, XER = 76%, WMT = 37%). Fifty-four percent of the Western Mountains ecoregion is characterized as forested. This high proportion of undeveloped land may be responsible for the higher IFI values seen in the three regions. Of the two regions with the lowest IFI values, 69% of the Temperate Plain ecoregion and 26% of the Coastal Plain is classified as cultivated/pasture. The high percentage of agricultural development reflected by this land use is potentially driving the lower IFI values seen in the region. The Coastal Plain ecoregion is made up of many different land use types, as it encompasses the Mississippi Delta, Gulf Coast, the entire state of Florida, a portion of eastern Texas, and the Atlantic coast from Florida to New Jersey. The large variety in land use types and geospatial

variability in the degree of anthropogenic modification seen through the Coastal Plain ecoregion may explain the low IFI values observed.

4.2 Limitations of IFI approach

The IFI methodology is limited by the availability, validity, and scope of the stressor datasets included in the index of integrity calculation. There are many other anthropogenic stressors that may exist within the floodplain not included in IFI methodology due to dataset unavailability or uncertainty driven by legacy effects. Such stressors may include the presence of pesticides, non-native vegetation, extirpation of beavers, historical removal of large wood, bank stabilization, and watershed land cover changes. The IFI methodology also only accounts for the presence of anthropogenic stressors within the floodplain itself and does not consider the impact of surrounding stressors that may impact floodplain health. The IFI methodology provides a broad quantitative assessment of floodplain health on a large-scale, but the limitations of the methods used must be acknowledged when assessing the results of the integrity index.

4.3 Trajectory of IFI from 1941 to 2000

Using the IFI methodology as a tool to develop a temporal trajectory of floodplain health over time was not as successful as I had anticipated. Ideally, the degree of change in IFI results over time would have been large enough to allow for a trajectorial analysis of floodplain health as driven by land cover changes. Variable data inputs were only used for two of the eleven total datasets that impact the calculation of overall floodplain integrity (see Section 2.4). This means that roughly 80% of the datasets used in the IFI methodology are remaining constant from one time step to the next, which may be a key contribution to the minimal temporal change observed. Additionally, the datasets that are not changing between iterations represent present-day stressor prevalence and do not accurately represent the historical stressor densities in the floodplain. I

knew this limitation when attempting to develop a trajectory of floodplain health, but I had hoped that the land cover data would provide enough variability to support a temporal analysis. It is also important to note that the dataset only accounts for land cover change within the floodplain itself, not the entire watershed. The total agricultural area in the Mississippi River Basin floodplain only increases by about 3.8% from 1941 to 2000. The total developed area in the floodplain more than doubles from 1941 to 2000, but it only accounts for about 1.9% of the floodplain in 1941 and 4.3% of the floodplain in 2000. I speculate that such low increases in agricultural and developed area density within the floodplain is a large factor in why little change is seen in the calculated IFI values over time. Many modifications to the river corridor along the Mississippi also began several decades before 1941 (Remo et al., 2009) and predate the earliest land cover estimate developed by (Rajib et al., 2021). The first discrete periods of engineering activity along the Mississippi began as early as the 1880s, with marked changes in hydrologic function noted shortly after (Collins and Knox, 2003; Pinter, 2005; Pinter and Heine, 2005). The minimal change seen in IFI results from 1941 to 2000 may be explained by the fact that large-scale anthropogenic alterations to the MRB began over a half-century prior to 1941.

Localized changes in IFI values occurring over the 60-year period can still be seen (Figure 13), although the variation in IFI results between each time step was not significant. The visualization of these localized changes indicates that the IFI methodology is still useful for directing attention to the areas experiencing alterations in floodplain functionality driven by human activity. Although the temporal dataset did not produce significantly different results between time steps, the use of the historical data still provided valuable insights into the adaptability of the IFI methodology and allowed for a discussion of its potential limitations. The most useful adaptation of the IFI methodology over time would likely need a robust set of

historical data, which is limited on the national scale. The application of the IFI methodology as a temporal tool would potentially be easier to produce on a smaller and more localized scale where historical data is available for most of the stressor datasets.

Our understanding of the trajectory of floodplain health over time may need to rely on a qualitative investigation of the historical impacts of anthropogenic activity, rather than a quantitative one. There is great complexity associated with analyzing the impact of historical human alterations to river systems. Wohl (2019) describes the challenges of understanding the implications of these legacy effects on watersheds and emphasizes the importance of using the history of river corridors as a tool to better inform the future of watershed management.

4.4 Considerations for future work

The application of the IFI methodology at the CONUS scale allowed for a better understanding of the spatial variability in floodplain health across the CONUS. However, a more robust understanding of floodplain health at the localized level will be necessary before explicitly selecting the floodplains most in need of restorative efforts. The CONUS IFI results are intended to be used as a tool to understand the heterogeneity in floodplain functionality for the nation as a whole and do not provide a detailed picture of floodplain health on a localized level. However, since the methodology was designed to be adaptable and flexible, it can be repeated at these smaller scales to analyze floodplain functionality at a finer resolution.

The IFI methodology is also based on a negative linear relationship between stressor density and critical floodplain function, which is an oversimplification of the complex relationship between anthropogenic development and floodplain health. The methodology could be revised with a more robust stressor density to floodplain health relationship (see Karpack et al., 2020). The type of relationship established between stressor prevalence and floodplain health

may change the range and magnitude of values reflected in the resulting index, but the IFI methodology was designed to be iteratively improved upon and revised in this manner. Finally, the temporal analysis of floodplain health established in this research provides the opportunity for additional investigation into the impact of human development on floodplain health over time.

5 Conclusion

This research develops an index of integrity for floodplain functionality at the national level using large-scale datasets. The IFI methodology was applied across the CONUS to gain a better understanding of the geospatial variability in floodplain health as indicated by human modifications. The applied methodology uses the density of human-induced stressors within the floodplain to develop a quantitative relationship between anthropogenic modification and floodplain functionality. A comprehensive evaluation of floodplain health is possible through this methodology since floodplain functionality is defined by a floodplain's ability to perform five critical functions (flood reduction, groundwater storage, sediment retention, organics and solutes retention, and habitat provision). The application of the IFI methodology to the CONUS successfully identified the spatial heterogeneity in floodplain functionality across the country. Overall IFI values tend to decrease with increasing stream order and in floodplains that intersect urban areas, providing further support for the fact that human development harms floodplain health. A variety of IFI distributions for both function and overall IFI values can be seen at the regional level, as shown by the three selected sub-regions in this study. Since the IFI methodology was applied at a broad scale, further work at a finer resolution will be needed to better understand floodplain health at the localized level. The dataset provides initial guidance for the floodplain areas in the US most impacted by anthropogenic activity and can act as a

preliminary guide for restoration efforts. The successful development of a temporal trajectory of floodplain health based on comprehensive historical datasets would grant a more robust understanding of floodplain health over time. A full analysis of floodplain health in the past will allow for a better-informed prediction of floodplain health in the future.

References

Acharya, G., 2000. Approaches to valuing the hidden hydrological services of wetland ecosystems. Ecol. Econ. 35, 63–74. https://doi.org/10.1016/S0921-8009(00)00168-3

Amoros, C., Bornette, G., 2002. Connectivity and biocomplexity in waterbodies of riverine floodplains. Freshw. Biol. 47, 761–776. https://doi.org/10.1046/j.1365-2427.2002.00905.x

Beevers, L., Douven, W., Lazuardi, H., Verheij, H., 2012. Cumulative impacts of road developments in floodplains. Transp. Res. Part Transp. Environ. 17, 398–404. https://doi.org/10.1016/j.trd.2012.02.005

Blanton, P., Marcus, W.A., 2009. Railroads, roads and lateral disconnection in the river landscapes of the continental United States. Geomorphology 112, 212–227. https://doi.org/10.1016/j.geomorph.2009.06.008

Böck, K., Polt, R., Schülting, L., 2018. Ecosystem Services in River Landscapes, in: Schmutz, S., Sendzimir, J. (Eds.), Riverine Ecosystem Management: Science for Governing Towards a Sustainable Future. Springer International Publishing, Cham, pp. 413–433. https://doi.org/10.1007/978-3-319-73250-3_21

Boulton, A.J., Datry, T., Kasahara, T., Mutz, M., Stanford, J.A., 2010. Ecology and management of the hyporheic zone: stream–groundwater interactions of running waters and their floodplains. J. North Am. Benthol. Soc. 29, 26–40. https://doi.org/10.1899/08-017.1

Brunke, M., Gonser, T., 1997. The ecological significance of exchange processes between rivers and groundwater. Freshw. Biol. 37, 1–33. https://doi.org/10.1046/j.1365-2427.1997.00143.x

Bunn, S.E., Arthington, A.H., 2002. Basic Principles and Ecological Consequences of Altered Flow Regimes for Aquatic Biodiversity. Environ. Manage. 30, 492–507. https://doi.org/10.1007/s00267-002-2737-0

Burt, T.P., 1997. The hydrological role of floodplains within the drainage basin system., in: Haycock, N. (Ed.), Buffer Zones: Their Processes and Potential in Water Protection ; the Proceedings of the International Conference on Buffer Zones, September 1996. Presented at the International Conference on Buffer Zones, Quest Environmental, Harpenden, pp. 21–32.

Collins, M.J., Knox, J.C., 2003. HISTORICAL CHANGES IN UPPER MISSISSIPPI RIVER WATER AREAS AND ISLANDS1. JAWRA J. Am. Water Resour. Assoc. 39, 487–500. https://doi.org/10.1111/j.1752-1688.2003.tb04401.x

Craig, L.S., Palmer, M.A., Richardson, D.C., Filoso, S., Bernhardt, E.S., Bledsoe, B.P., Doyle, M.W., Groffman, P.M., Hassett, B.A., Kaushal, S.S., Mayer, P.M., Smith, S.M., Wilcock, P.R., 2008. Stream restoration strategies for reducing river nitrogen loads. Front. Ecol. Environ. 6, 529–538. https://doi.org/10.1890/070080

DeBoer, J.A., Thoms, M.C., Delong, M.D., 2022. Ecosystem Response Through a Resilience Lens: Do Differences in the Illinois River Over 150 Y Indicate Regime Shifts? J. Geophys. Res. Biogeosciences 127, e2021JG006553. https://doi.org/10.1029/2021JG006553

Dewitz, J., 2021. National Land Cover Database (NLCD) 2019 Products. https://doi.org/10.5066/P9KZCM54

ERSI, 2011. ArcGIS Desktop: Release 10.

Erwin, K.L., 2008. Wetlands and global climate change: the role of wetland restoration in a changing world. Wetl. Ecol. Manag. 17, 71. https://doi.org/10.1007/s11273-008-9119-1
Falcone, J.A., 2017. U.S. Geological Survey GAGES-II time series data from consistent sources of land use, water use, agriculture, timber activities, dam removals, and other historical anthropogenic influences. https://doi.org/10.5066/F7HQ3XS4
Farnsworth, K.L., Milliman, J.D., 2003. Effects of climatic and anthropogenic change on small mountainous rivers: the Salinas River example. Glob. Planet. Change, The supply of flux of sediment along hydrological pathways: Anthropogenic influences at the global scale 39, 53–64. https://doi.org/10.1016/S0921-8181(03)00017-1
Fischer, C., Damm, C., Foeckler, F., Gelhaus, M., Gerstner, L., Harris, R.M.B., Hoffmann, T.G., Iwanowski, J., Kasperidus, H., Mehl, D., Podschun, S.A., Rumm, A., Stammel, B., Scholz, M., 2019. The "Habitat Provision" Index for Assessing Floodplain Biodiversity and Restoration Potential as an Ecosystem Service—Method and Application. Front. Ecol. Evol. 7. https://doi.org/10.3389/fevo.2019.00483
Flotemersch, J.E., Leibowitz, S.G., Hill, R.A., Stoddard, J.L., Thoms, M.C., Tharme, R.E., 2016. A Watershed Integrity Definition and Assessment Approach to Support Strategic Management of Watersheds. River Res. Appl. 32, 1654–1671. https://doi.org/10.1002/rra.2978
Fryirs, K., 2013. (Dis)Connectivity in catchment sediment cascades: a fresh look at the sediment delivery problem. Earth Surf. Process. Landf. 38, 30–46. https://doi.org/10.1002/esp.3242
Fuller, I.C., Death, R.G., 2018. The science of connected ecosystems: What is the role of catchment-scale connectivity for healthy river ecology? Land Degrad. Dev. 29, 1413–1426. https://doi.org/10.1002/ldr.2903
Graf, W.L., 2001. Damage Control: Restoring the Physical Integrity of America's Rivers. Ann. Assoc. Am. Geogr. 91, 1–27. https://doi.org/10.1111/0004-5608.00231
Habersack, H., Schober, B., Hauer, C., 2015. Floodplain evaluation matrix (FEM): An interdisciplinary method for evaluating river floodplains in the context of integrated flood risk management. Nat. Hazards 75, 5–32. https://doi.org/10.1007/s11069-013-0842-4
Hanberry, B.B., Kabrick, J.M., He, H.S., 2015. Potential tree and soil carbon storage in a major historical floodplain forest with disrupted ecological function. Perspect. Plant Ecol. Evol. Syst. 17, 17–23. https://doi.org/10.1016/j.ppees.2014.12.002
Hansen M. C., Potapov P. V., Moore R., Hancher M., Turubanova S. A., Tyukavina A., Thau D., Stehman S. V., Goetz S. J., Loveland T. R., Kommareddy A., Egorov A., Chini L., Justice C. O., Townshend J. R. G., 2013. High-Resolution Global Maps of 21st-Century Forest Cover Change. Science 342, 850–853. https://doi.org/10.1126/science.1244693
Harper, D., Mekotova, J., Hulme, S., White, J., Hall, J., 1997. Habitat Heterogeneity and Aquatic Invertebrate Diversity in Floodplain Forests. Glob. Ecol. Biogeogr. Lett. 6, 275–285. https://doi.org/10.2307/2997741
Herlihy, A.T., Paulsen, S.G., Sickle, J.V., Stoddard, J.L., Hawkins, C.P., Yuan, L.L., 2008. Striving for consistency in a national assessment: the challenges of applying a reference-condition approach at a continental scale. J. North Am. Benthol. Soc. 27, 860–877. https://doi.org/10.1899/08-081.1
Hinshaw, S., Wohl, E., Burnett, J.D., Wondzell, S., 2022. Development of a geomorphic monitoring strategy for stage 0 restoration in the South Fork McKenzie River, Oregon, USA. Earth Surf. Process. Landf. n/a. https://doi.org/10.1002/esp.5356

Hopkins, K.G., Noe, G.B., Franco, F., Pindilli, E.J., Gordon, S., Metes, M.J., Claggett, P.R., Gellis, A.C., Hupp, C.R., Hogan, D.M., 2018. A method to quantify and value floodplain sediment and nutrient retention ecosystem services. J. Environ. Manage. 220, 65–76. https://doi.org/10.1016/j.jenvman.2018.05.013

Hupp, C.R., Pierce, A.R., Noe, G.B., 2009. Floodplain geomorphic processes and environmental impacts of human alteration along Coastal Plain rivers, USA. Wetlands 29, 413–429. https://doi.org/10.1672/08-169.1

Johnson, K.A., Wing, O.E.J., Bates, P.D., Fargione, J., Kroeger, T., Larson, W.D., Sampson, C.C., Smith, A.M., 2020. A benefit–cost analysis of floodplain land acquisition for US flood damage reduction. Nat. Sustain. 3, 56–62. https://doi.org/10.1038/s41893-019-0437-5

Johnson, Z.C., Leibowitz, S.G., Hill, R.A., 2019. Revising the index of watershed integrity national maps. Sci. Total Environ. 651, 2615–2630. https://doi.org/10.1016/j.scitotenv.2018.10.112

Junk, W., Bayley, P., Sparks, R., 1989. The Flood Pulse Concept in River-Floodplain Systems, Can. Spec. Public Fish. Aquat. Sci.

Karpack, M.N., Morrison, R.R., McManamay, R.A., 2020. Quantitative assessment of floodplain functionality using an index of integrity. Ecol. Indic. 111, 106051. https://doi.org/10.1016/j.ecolind.2019.106051

Karr, J.R., 1981. Assessment of Biotic Integrity Using Fish Communities. Fisheries 6, 21–27. https://doi.org/10.1577/1548-8446(1981)006<0021:AOBIUF>2.0.CO;2

Kesel, R.H., 2003. Human modifications to the sediment regime of the Lower Mississippi River flood plain. Floodplains Environ. Process 56, 325–334. https://doi.org/10.1016/S0169-555X(03)00159-4

Knox, J.C., 2006. Floodplain sedimentation in the Upper Mississippi Valley: Natural versus human accelerated. Geomorphology, 37th Binghamton Geomorphology Symposium 79, 286–310. https://doi.org/10.1016/j.geomorph.2006.06.031

Knox, R.L., Morrison, R.R., Wohl, E., 2022. A river ran through it: Floodplains as America's newest relict landform.

Knox, R. L., Morrison, R.R., Wohl, E.E., 2022. Identification of Artificial Levees in the Contiguous United States. Water Resour. Res. 58, e2021WR031308. https://doi.org/10.1029/2021WR031308

Konrad, C.P., 2003. Effects of Urban Development on Floods. Fact Sheet. https://doi.org/10.3133/fs07603

Kousky, C., Walls, M., 2014. Floodplain conservation as a flood mitigation strategy: Examining costs and benefits. Ecol. Econ. 104, 119–128. https://doi.org/10.1016/j.ecolecon.2014.05.001

Loomis, J., Kent, P., Strange, L., Fausch, K., Covich, A., 2000. Measuring the total economic value of restoring ecosystem services in an impaired river basin: results from a contingent valuation survey. Ecol. Econ. 33, 103–117. https://doi.org/10.1016/S0921-8009(99)00131-7

McManamay, R., George, R., Morrison, R.R., Ruddell, B., In Review. Mapping Hydrologic Alteration and Ecological Consequences in Stream Reaches of the Conterminous United States. Sci. Data.

Moore, Richard B, Johnston, C.M., Hayes, L., 2019. Crosswalk Table Between NHDPlus V2.1 and its Accompanying Watershed Boundary Dataset Snapshot of 12-Digit Hydrologic Units. https://doi.org/10.5066/P9CFXHGT
Moore, Richard B., McKay, L.D., Rea, A.H., Bondelid, T.R., Price, C.V., Dewald, T.G., Johnston, C.M., 2019. User's guide for the national hydrography dataset plus (NHDPlus) high resolution. Users Guide Natl. Hydrogr. Dataset Plus NHDPlus High Resolut., Open-File Report 2019–1096, 80. https://doi.org/10.3133/ofr20191096
Mossa, J., 1996. Sediment dynamics in the lowermost Mississippi River. Eng. Geol. 45, 457–479. https://doi.org/10.1016/S0013-7952(96)00026-9
Munné, A., Prat, N., Solà, C., Bonada, N., Rieradevall, M., 2003. A simple field method for assessing the ecological quality of riparian habitat in rivers and streams: QBR index. Aquat. Conserv. Mar. Freshw. Ecosyst. 13, 147–163. https://doi.org/10.1002/aqc.529
Nilsson, C., Reidy, C.A., Dynesius, M., Revenga, C., 2005. Fragmentation and Flow Regulation of the World's Large River Systems. Science 308, 405–408. https://doi.org/10.1126/science.1107887
Noe, G.B., Hupp, C.R., 2009. Retention of Riverine Sediment and Nutrient Loads by Coastal Plain Floodplains. Ecosystems 12, 728–746. https://doi.org/10.1007/s10021-009-9253-5
Olde Venterink, H., Vermaat, J.E., Pronk, M., Wiegman, F., van der Lee, G.E.M., van den Hoorn, M.W., Higler, L.W.G. (Bert), Verhoeven, J.T.A., 2006. Importance of sediment deposition and denitrification for nutrient retention in floodplain wetlands. Appl. Veg. Sci. 9, 163–174. https://doi.org/10.1111/j.1654-109X.2006.tb00665.x
Olden, J.D., Poff, N.L., 2003. Redundancy and the choice of hydrologic indices for characterizing streamflow regimes. River Res. Appl. 19, 101–121. https://doi.org/10.1002/rra.700
Open Street Map Contributors, 2022. GEOFABRIK // Shapefiles [WWW Document]. URL http://www.geofabrik.de/data/shapefiles.html (accessed 5.2.22).
Opperman, J.J., Galloway, G.E., Fargione, J., Mount, J.F., Richter, B.D., Secchi, S., 2009. Sustainable Floodplains Through Large-Scale Reconnection to Rivers. Science 326, 1487–1488. https://doi.org/10.1126/science.1178256
Opperman, J.J., Moyle, P.B., Larsen, E.W., Florsheim, J.L., Manfree, A.D., 2017. Introduction to Temperate Floodplains, in: Floodplains: Processes and Management for Ecosystem Services. University of California Press, Berkeley, UNITED STATES, pp. 1–10.
Peipoch, M., Brauns, M., Hauer, F.R., Weitere, M., Valett, H.M., 2015. Ecological Simplification: Human Influences on Riverscape Complexity. BioScience 65, 1057–1065. https://doi.org/10.1093/biosci/biv120
Pinay, G., Clément, J.C., Naiman, R.J., 2002. Basic Principles and Ecological Consequences of Changing Water Regimes on Nitrogen Cycling in Fluvial Systems. Environ. Manage. 30, 481–491. https://doi.org/10.1007/s00267-002-2736-1
Pinter, N., 2005. One Step Forward, Two Steps Back on U.S. Floodplains. Science 308, 207–208. https://doi.org/10.1126/science.1108411
Pinter, N., Heine, R.A., 2005. Hydrodynamic and morphodynamic response to river engineering documented by fixed-discharge analysis, Lower Missouri River, USA. J. Hydrol. 302, 70–91. https://doi.org/10.1016/j.jhydrol.2004.06.039
Rajib, A., Zheng, Q., Golden, H.E., Wu, Q., Lane, C.R., Christensen, J.R., Morrison, R.R., Annis, A., Nardi, F., 2021. The changing face of floodplains in the Mississippi River

Basin detected by a 60-year land use change dataset. Sci. Data 8, 271. https://doi.org/10.1038/s41597-021-01048-w
Remo, J.W.F., Ickes, B.S., Ryherd, J.K., Guida, R.J., Therrell, M.D., 2018. Assessing the impacts of dams and levees on the hydrologic record of the Middle and Lower Mississippi River, USA. Geomorphology 313, 88–100. https://doi.org/10.1016/j.geomorph.2018.01.004
Remo, J.W.F., Pinter, N., Heine, R., 2009. The use of retro- and scenario-modeling to assess effects of 100+ years river of engineering and land-cover change on Middle and Lower Mississippi River flood stages. J. Hydrol. 376, 403–416. https://doi.org/10.1016/j.jhydrol.2009.07.049
Rollins, M.G., 2009. LANDFIRE: a nationally consistent vegetation, wildland fire, and fuel assessment. Int. J. Wildland Fire 18, 235. https://doi.org/10.1071/WF08088
RStudio Team, 2015. RStudio: Integrated Development Environment for R. RStudio, Inc., Boston, MA.
Schober, B., Hauer, C., Habersack, H., 2015. A novel assessment of the role of Danube floodplains in flood hazard reduction (FEM method). Nat. Hazards 75, 33–50. https://doi.org/10.1007/s11069-013-0880-y
Seaber, P.R., Kapinos, F.P., Knapp, G.L., 1987. Hydrologic unit maps (USGS Numbered Series No. 2294), Hydrologic unit maps, Water Supply Paper. U.S. G.P.O.,. https://doi.org/10.3133/wsp2294
Sharma, S., 2017. Geological Behaviours in Urban Areas for Surface Runoff and Recharge. Open J. Soil Sci. 7, 181–201. https://doi.org/10.4236/ojss.2017.78014
Smucker, N.J., Detenbeck, N.E., 2014. Meta-Analysis of Lost Ecosystem Attributes in Urban Streams and the Effectiveness of Out-of-Channel Management Practices. Restor. Ecol. 22, 741–748. https://doi.org/10.1111/rec.12134
Snyder, E.B., Arango, C.P., Eitemiller, D.J., Stanford, J.A., Uebelacker, M.L., 2002. Floodplain hydrologic connectivity and fisheries restoration in the Yakima River, U.S.A. SIL Proc. 1922-2010 28, 1653–1657. https://doi.org/10.1080/03680770.2001.11901901
Stoddard, J.L., Larsen, D.P., Hawkins, C.P., Johnson, R.K., Norris, R.H., 2006. Setting Expectations for the Ecological Condition of Streams: The Concept of Reference Condition. Ecol. Appl. 16, 1267–1276. https://doi.org/10.1890/1051-0761(2006)016[1267:SEFTEC]2.0.CO;2
Stone, M.C., Byrne, C.F., Morrison, R.R., 2017. Evaluating the impacts of hydrologic and geomorphic alterations on floodplain connectivity. Ecohydrology 10, e1833. https://doi.org/10.1002/eco.1833
Sun, R., Yao, P., Wang, W., Yue, B., Liu, G., 2017. Assessment of Wetland Ecosystem Health in the Yangtze and Amazon River Basins. ISPRS Int. J. Geo-Inf. 6, 81. https://doi.org/10.3390/ijgi6030081
Sutfin, N.A., Wohl, E.E., Dwire, K.A., 2016. Banking carbon: a review of organic carbon storage and physical factors influencing retention in floodplains and riparian ecosystems. Earth Surf. Process. Landf. 41, 38–60. https://doi.org/10.1002/esp.3857
Tarolli, P., Sofia, G., 2016. Human topographic signatures and derived geomorphic processes across landscapes. Geomorphology 255, 140–161. https://doi.org/10.1016/j.geomorph.2015.12.007
Thornbrugh, D.J., Leibowitz, S.G., Hill, R.A., Weber, M.H., Johnson, Z.C., Olsen, A.R., Flotemersch, J.E., Stoddard, J.L., Peck, D.V., 2018. Mapping watershed integrity for the

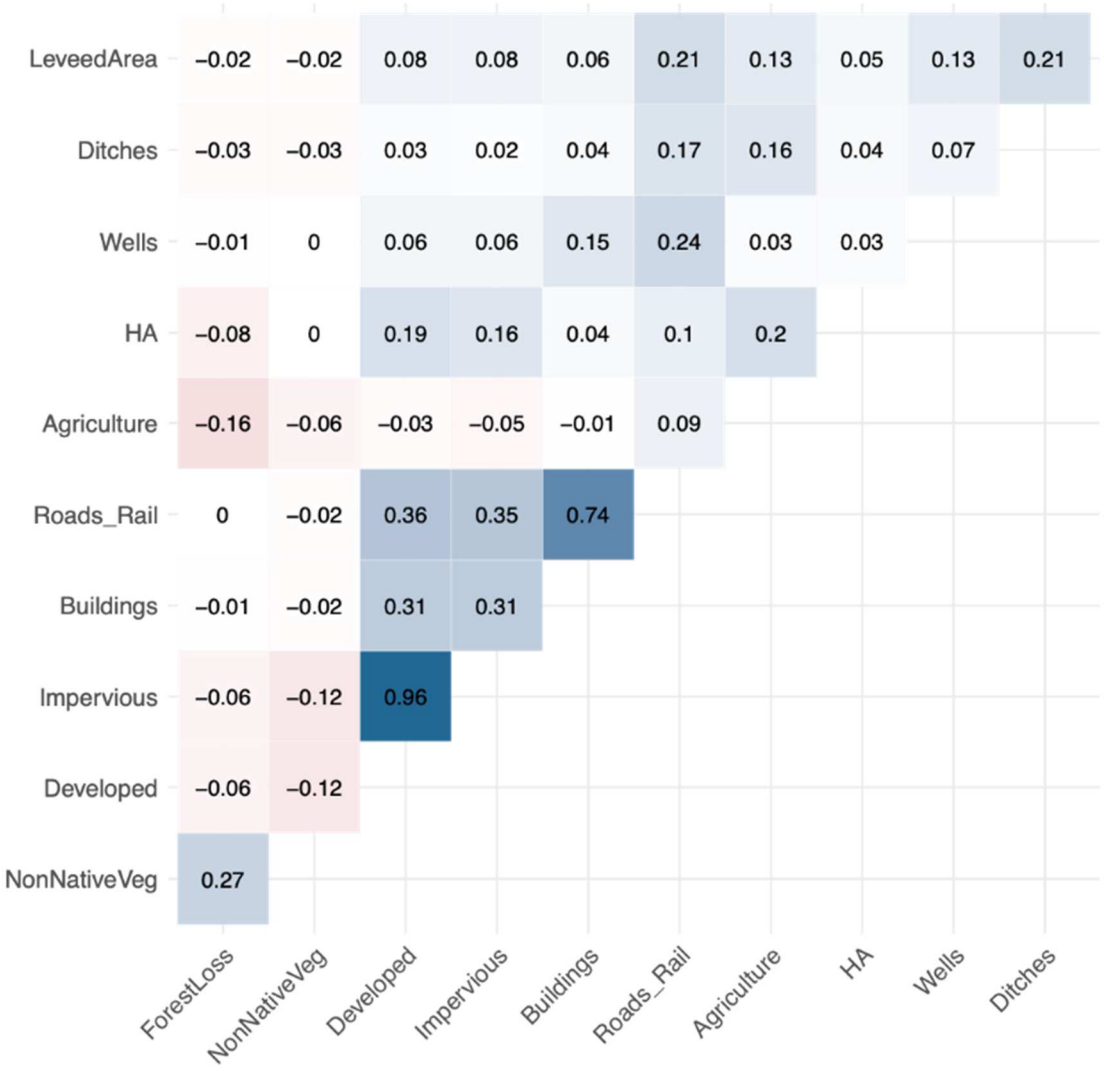
conterminous United States. Ecol. Indic. 85, 1133–1148. https://doi.org/10.1016/j.ecolind.2017.10.070
Tockner, K., Pennetzdorfer, D., Reiner, N., Schiemer, F., Ward, J.V., 1999. Hydrological connectivity, and the exchange of organic matter and nutrients in a dynamic river–floodplain system (Danube, Austria). Freshw. Biol. 41, 521–535. https://doi.org/10.1046/j.1365-2427.1999.00399.x
Tockner, K., Stanford, J.A., 2002. Riverine flood plains: present state and future trends. Environ. Conserv. 29, 308–330. https://doi.org/10.1017/S037689290200022X
Tomscha, S.A., Gergel, S.E., Tomlinson, M.J., 2017. The spatial organization of ecosystem services in river-floodplains. Ecosphere 8, e01728. https://doi.org/10.1002/ecs2.1728
Tukey, J.W., 1953. The problem of multiple comparisons.
U.S. Army Corps of Engineers, 2019. National Levee Database [WWW Document]. URL https://levees.sec.usace.army.mil/#/ (accessed 5.2.22).
US Census Bureau, 2019. TIGER/Line Shapefiles [WWW Document]. Census.gov. URL https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html (accessed 5.2.22).
US EPA, 2020. National Rivers and Streams Assessment 2013-2014 [WWW Document]. US EPA. URL http://www.epa.gov/national-aquatic-resource-surveys/data-national-aquatic-resource-surveys.
US EPA, 2016. Ecoregions used in the National Aquatic Resource Surveys [WWW Document]. URL https://www.epa.gov/national-aquatic-resource-surveys/ecoregions-used-national-aquatic-resource-surveys (accessed 5.3.22).
U.S. Geological Survey, 2022. USGS -- Groundwater Watch [WWW Document]. URL https://groundwaterwatch.usgs.gov/default.asp (accessed 5.2.22).
Wade, T.G., Ebert, D.W., 2016. ANALYTICAL TOOLS INTERFACE FOR LANDSCAPE ASSESSMENTS (ATTILA) USER MANUAL.
Ward, J.V., 1989. The Four-Dimensional Nature of Lotic Ecosystems. J. North Am. Benthol. Soc. 8, 2–8. https://doi.org/10.2307/1467397
Ward, J.V., Stanford, J.A., 1995. The serial discontinuity concept: Extending the model to floodplain rivers. Regul. Rivers Res. Manag. 10, 159–168. https://doi.org/10.1002/rrr.3450100211
Ward, J.V., Tockner, K., Schiemer, F., 1999. Biodiversity of floodplain river ecosystems: ecotones and connectivity1. Regul. Rivers Res. Manag. 15, 125–139. https://doi.org/10.1002/(SICI)1099-1646(199901/06)15:1/3<125::AID-RRR523>3.0.CO;2-E
Wheater, H., Evans, E., 2009. Land use, water management and future flood risk. Land Use Policy, Land Use Futures 26, S251–S264. https://doi.org/10.1016/j.landusepol.2009.08.019
Wing, O.E.J., Bates, P.D., Sampson, C.C., Smith, A.M., Johnson, K.A., Erickson, T.A., 2017. Validation of a 30 m resolution flood hazard model of the conterminous United States. Water Resour. Res. 53, 7968–7986. https://doi.org/10.1002/2017WR020917
Wohl, E., 2021. An Integrative Conceptualization of Floodplain Storage. Rev. Geophys. 59, e2020RG000724. https://doi.org/10.1029/2020RG000724
Wohl, E., 2019. Forgotten Legacies: Understanding and Mitigating Historical Human Alterations of River Corridors. Water Resour. Res. 55, 5181–5201. https://doi.org/10.1029/2018WR024433

Wohl, E., Bledsoe, B.P., Jacobson, R.B., Poff, N.L., Rathburn, S.L., Walters, D.M., Wilcox, A.C., 2015. The Natural Sediment Regime in Rivers: Broadening the Foundation for Ecosystem Management. BioScience 65, 358–371. https://doi.org/10.1093/biosci/biv002

Appendix A: Additional Analyses

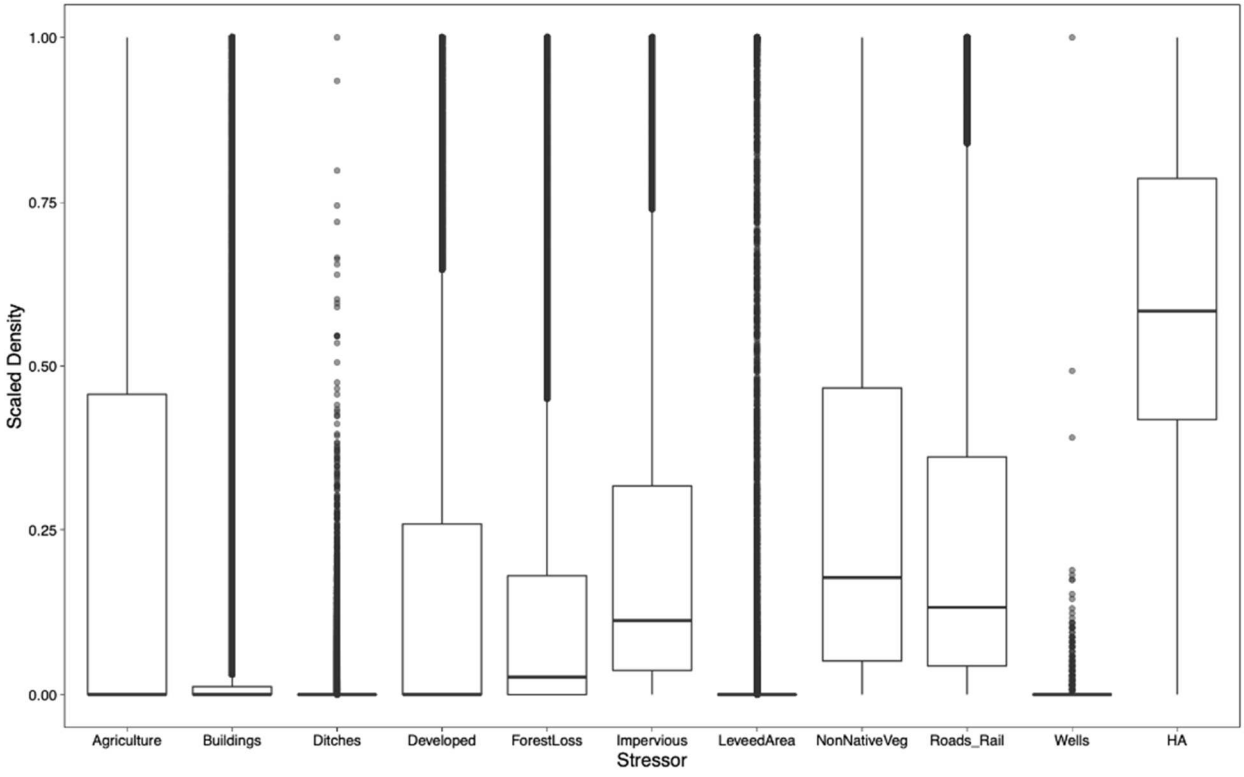
Stressor dataset correlation

If the Pearson's correlation coefficient between any two stressors exceeded 0.7, only one stressor dataset was used in the IFI calculation. This was done to avoid overweighing any one individual stressor. The correlation matrix between all stressor datasets can be seen below.



Scaled stressor dataset densities

The density of the scaled stressor datasets in the CONUS can be seen below.



Example function and overall IFI calculation

An example calculation of how individual function IFI values and overall IFI for one floodplain unit within the Mississippi Yazoo region (HUC12 code 080302071701) is summarized below.

Stressor Dataset	Scaled Density
Agriculture	0.000
Buildings	0.000
Ditches	0.064
Developed	0.272
ForestLoss	0.061
Impervious	0.262
LeveedArea	0.999
NonNativeVeg	0.191
Roads_Rail	1.000
Wells	0.000
HA	0.515

Flood reduction: Developed, ForestLoss, LeveedArea, Roads_Rail (Buildings were excluded from the flood reduction IFI calculation due to high correlation and to avoid overweighting.)

$$IFI_{floods} = 1 - \frac{0.999 + 1 + 0.061 + 0.272}{4} = 1 - 0.583 = 0.417$$

Groundwater storage: Agriculture, Ditches, ForestLoss, Impervious, Wells

$$IFI_{groundwater} = 1 - \frac{0 + 0.064 + 0.061 + 0.262 + 0}{5} = 1 - 0.077 = 0.923$$

Sediment regulation: Agriculture, ForestLoss, Roads_Rail, HA

$$IFI_{sediment} = 1 - \frac{0 + 0.061 + 1 + 0.515}{4} = 1 - 0.394 = 0.606$$

Organics and solute regulation: ForestLoss, Impervious, HA, Roads_Rail

$$IFI_{organics.solutes} = 1 - \frac{0.061 + 0.262 + 1 + 0.515}{4} = 1 - 0.459 = 0.541$$

Habitat provision: Agriculture, Developed, ForestLoss, NonNativeVeg, Roads_Rail, HA

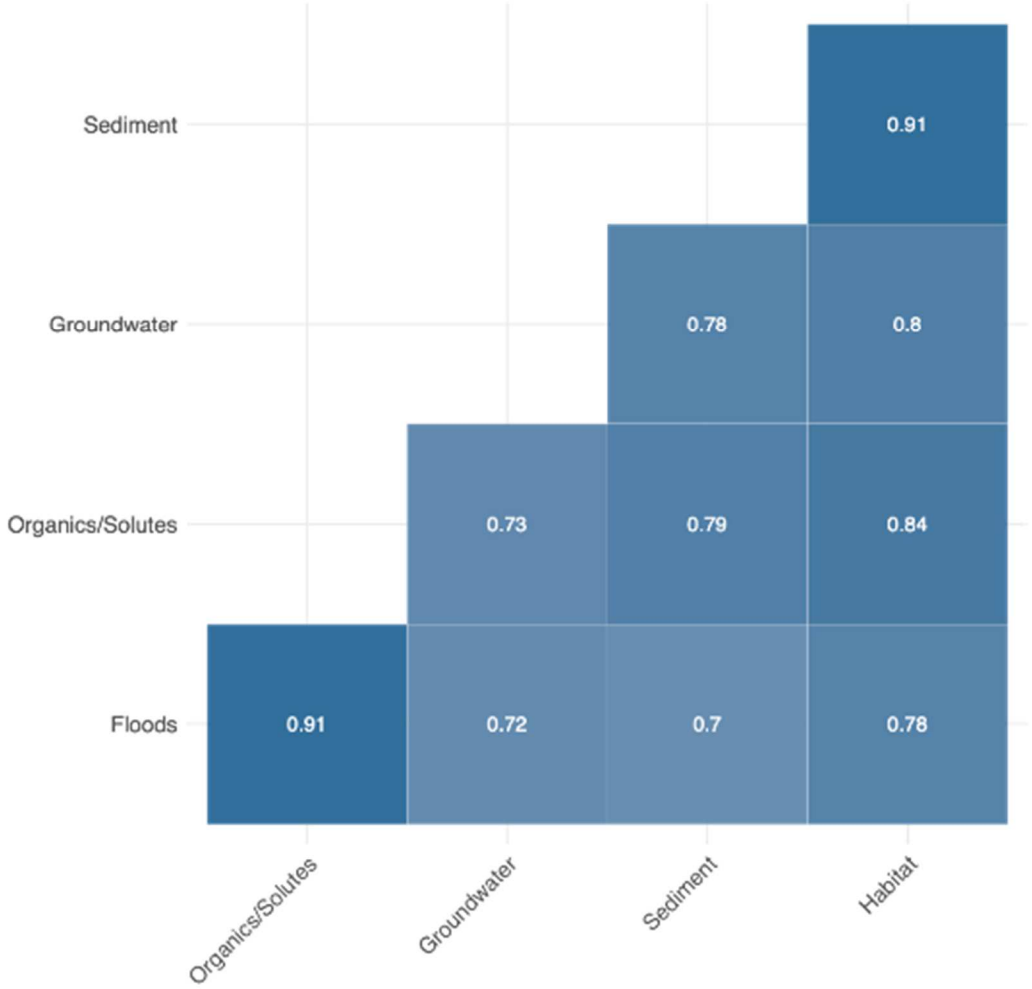
$$IFI_{habitat} = 1 - \frac{0 + 0.272 + 0.061 + 0.191 + 1 + 0.515}{6} = 1 - 0.340 = 0.660$$

Overall IFI is calculated by taking the geometric mean of the function IFI values

$$IFI_{overall} = (0.417 * 0.923 * 0.606 * 0.541 * 0.660)^{\frac{1}{5}} = 0.608$$

Correlation of function IFI values

A correlation matrix using Pearson's correlation coefficients for the floodplain function IFI values can be seen below.



Selected subregion statistics

Summary statistics of overall and function integrity values for San Joaquin sub-region

	min	median	mean	max	sd
Flood Reduction	0.159	0.836	0.782	1.000	0.201
Groundwater Storage	0.413	0.873	0.832	1.000	0.150
Sediment Regulation	0.015	0.720	0.629	0.953	0.244
Organics/Solutes Regulation	0.079	0.672	0.627	0.940	0.211
Habitat Provision	0.119	0.705	0.644	0.954	0.208
Overall	0.148	0.742	0.691	0.957	0.204

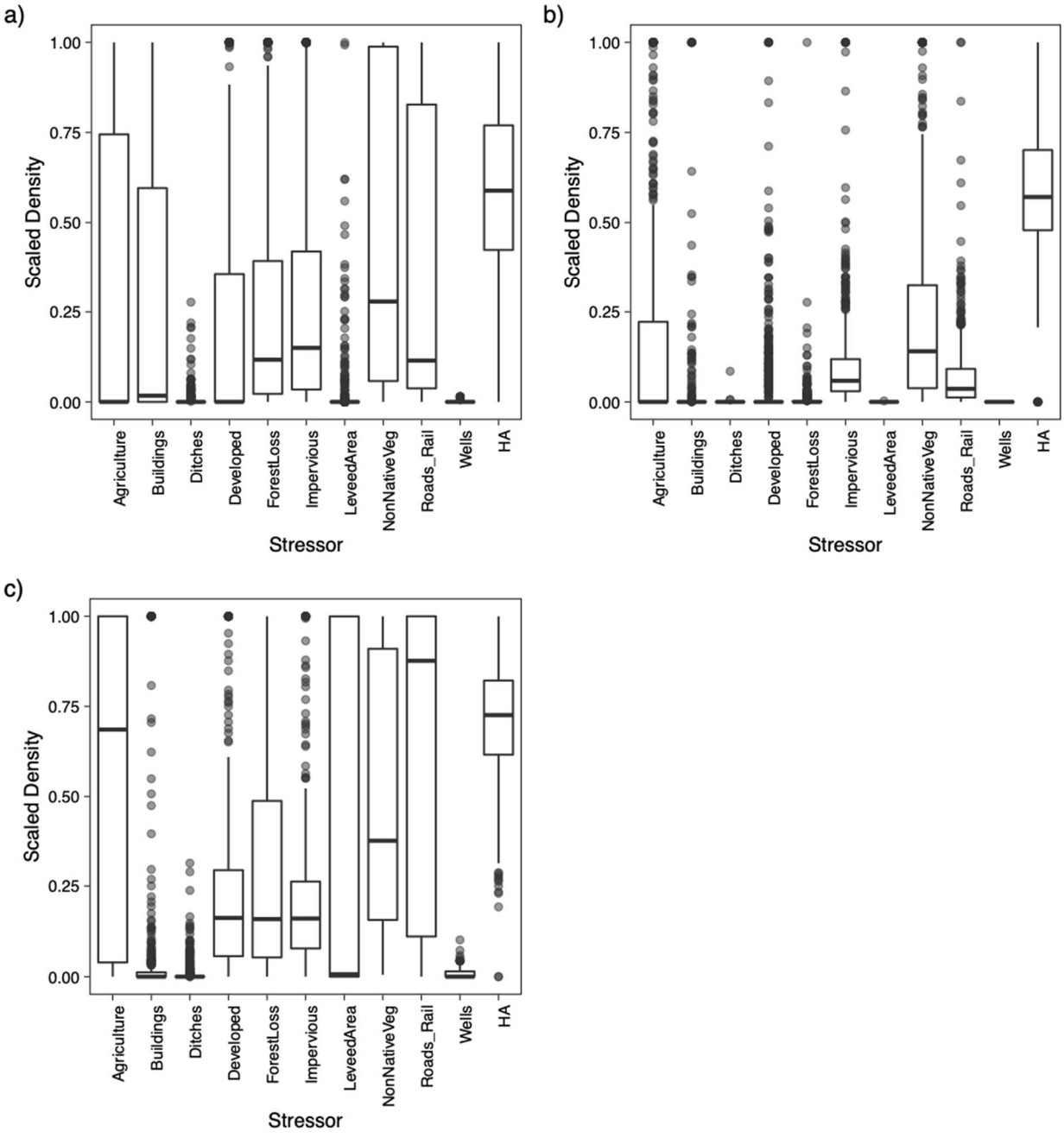
Summary statistics of overall and function integrity values for Missouri-White sub-region

	min	median	mean	max	sd
Flood Reduction	0.476	0.987	0.968	1.000	0.056
Groundwater Storage	0.733	0.972	0.944	1.000	0.065
Sediment Regulation	0.463	0.811	0.791	0.999	0.090
Organics/Solutes Regulation	0.226	0.819	0.808	0.991	0.073
Habitat Provision	0.441	0.835	0.813	0.962	0.088
Overall	0.443	0.874	0.860	0.974	0.065

Summary statistics of overall and function integrity values for Lower Mississippi-Yazoo sub-region

	min	median	mean	max	sd
Flood Reduction	0.210	0.627	0.616	1.000	0.215
Groundwater Storage	0.499	0.746	0.772	0.990	0.114
Sediment Regulation	0.126	0.464	0.451	0.893	0.157
Organics/Solutes Regulation	0.099	0.518	0.539	0.889	0.145
Habitat Provision	0.164	0.506	0.517	0.881	0.125
Overall	0.223	0.557	0.561	0.914	0.131

The scaled stressor densities for a) San Joaquin, b) Missouri-White, and c) Lower Mississippi-Yazoo HUC4 subregions can be seen below.



Mississippi River Basin land use reclassifications and original land cover type

MRB Land Cover Class	Original Land Cover Type
Developed area	Mechanically disturbed national forests
	Mechanically disturbed other public lands
	Mining urban areas
Agriculture	Cropland
	Herbaceous cover
	Tree or shrub cover
	Cropland irrigated or post-flooding
	Mosaic cropland (>50%) /natural vegetation (tree shrub herbaceous cover) (<50%)
	Mosaic natural vegetation (tree shrub herbaceous cover) (>50%) /cropland (<50%)

Appendix B: R Coding Scripts

Stressor Datasets

The following R scripts were used to organize stressor datasets and calculate stressor densities within the floodplain:

```
###
# primary processing script for running content
#Kira Simonson, kira.simonson@colostate.edu
# CSU Geospatial Centroid, carverd@colostate.edu
# Spring 2022


###

# install.packages("pacman")
pacman::p_load(sf,terra,dplyr, tictoc, readr, purrr)

# source scripts ------------------------------------------------------
-
for(feature in list.files(path = "src/" ,full.names = TRUE)){
  source(feature)
}


# Calculate Stressor Dataset Density ----------------------------------
-
## reading from a local file, taking a long time to load pulling from
floodAreas <- sf::st_read("US_HU12/US_HU12.shp")

# OpenStreetMap -------------------------------------------------------
-
states <- st_read("Z:/National Data
Downloads/ne_10m_admin_1_states_provinces/ne_10m_admin_1_states_provinces.
shp")%>%
  dplyr::filter(adm0_a3 == "USA")%>%
  st_transform(crs = st_crs(floodAreas))%>%
  dplyr::select("postal")
# this takes a long time so I should write it out
s2 <- st_intersection(x = floodAreas, y = states)
# Join results to flood area file
st_postal <- s2 %>%
  st_drop_geometry()%>%
  group_by(FIRST_huc1)%>%
  dplyr::mutate(postal1 =paste0(postal, collapse = ", "))%>%
  dplyr::select(FIRST_huc1, postal1)
write_csv(x = st_postal, file = "US_HU12/US_HU12)stateIntersection.csv")

st_postal <- read_csv("US_HU12/US_HU12)stateIntersection.csv")

floodAreas2 <- dplyr::left_join(floodAreas, st_postal)
floodAreas2 <- floodAreas2[!duplicated(floodAreas2$FIRST_huc1), ]
```

```
### spilt out states to run on specific PCS
abbrev <- c("AL",
            "AZ","AR","CA","CO","CT","DE","FL","GA","ID",
            "IL","IN","IA","KS","KY","LA","ME","MD","MA","MI",
            "MN","MS","MO","MT","NE","NV","NH","NJ","NM",
            "NY","NC","ND","OH","OK","OR","PA","RI","SC","SD",
            "TN","TX","UT","VT","VA","WA","WV","WI","WY")

# search for completed files
sort(list.files(path = "outputs/", pattern = "TX", recursive = TRUE ))

completed <- c("AL",
            "AZ","AR",
            "ID",
            "IL","IN","IA","KS","KY","LA","ME","MD","MA","MI",
            "MN","MS","MO","MT","NE","NV","NH","NJ","NM",
            "NY","NC","ND","OH","OK","OR","PA"
)
# list of content to rerun,
abbrev2 <- abbrev[!abbrev %in% completed]
abbrev3 <- abbrev2[!abbrev2 %in% c("CA","TX")]
# spilt for building and rr completed
build3 <- abbrev3[abbrev3 !="TN"]
rr3 <- abbrev3

for(i in abbrev3){
  print(i)
  print(sort(list.files(path = "outputs/", pattern = i, recursive = TRUE
)))
}

geo3 <- build3[1:5]
geo4 <- build3[6:10]
geo5 <- build3[11:15]

## render process for each pc
# geo8 - testing
get_buildings(floodPlain =floodAreas2, stateAbbrev = geo8, outputFolder =
"outputs/osm/buildings" )
get_r_r(floodPlain =floodAreas2, stateAbbrev = geo8, outputFolder =
"outputs/osm/roadsRails")
# geo2 -- completed 20220304
get_buildings(floodPlain =floodAreas2, stateAbbrev = geo2, outputFolder =
"outputs/osm/buildings" )
get_r_r(floodPlain =floodAreas2, stateAbbrev = geo2, outputFolder =
"outputs/osm/roadsRails" )
# geo3
get_buildings(floodPlain =floodAreas2, stateAbbrev = geo3, outputFolder =
"outputs/osm/buildings" )
  get_r_r(floodPlain =floodAreas2, stateAbbrev = geo3, outputFolder =
"outputs/osm/roadsRails" )
# geo4
get_buildings(floodPlain =floodAreas2, stateAbbrev = geo4, outputFolder =
"outputs/osm/buildings" )
```

```
get_r_r(floodPlain =floodAreas2, stateAbbrev = geo4, outputFolder =
"outputs/osm/roadsRails" )
# geo5
get_buildings(floodPlain =floodAreas2, stateAbbrev = geo5, outputFolder =
"outputs/osm/buildings" )
get_r_r(floodPlain =floodAreas2, stateAbbrev = geo5, outputFolder =
"outputs/osm/roadsRails" )
# geo6
get_buildings(floodPlain =floodAreas2, stateAbbrev = geo6, outputFolder =
"outputs/osm/buildings" )
get_r_r(floodPlain =floodAreas2, stateAbbrev = geo6, outputFolder =
"outputs/osm/roadsRails" )


build1 <- compile(buildings = TRUE, path = "outputs/")
write_csv(x = build1, file = "outputs/compiledOutputs/buildings.csv")
rr_1 <- compile(buildings = FALSE, path = "outputs/")
write_csv(x = rr_1, file = "outputs/compiledOutputs/roads_rails.csv")

### test for missing records
build1 <- read_csv("outputs/osm/compiledOutputs/buildings.csv")
fp2 <- floodAreas[!floodAreas$FIRST_huc1 %in% build1$FIRST_huc1, ]
# are unclass hucs in state intersection? No, so edit OSM code to run
intersection
states <- st_read("National Data
Downloads/cb_2018_us_state_500k/cb_2018_us_state_500k.shp")%>%
  st_transform(crs = st_crs(floodAreas))%>%
  dplyr::select("STUSPS")

# this takes a long time so I should write it out
s2 <- st_intersection(x = fp2, y = states)

fp4 <- fp2[!fp2$FIRST_huc1 %in% s2$FIRST_huc1, ]
write_csv(x = fp4, file = "US_HU12/US_HU12_noStates_rr.csv")

# Join results to flood area file
st_postal <- s2 %>%
  st_drop_geometry()%>%
  group_by(FIRST_huc1)%>%
  dplyr::select(FIRST_huc1, postal1 = STUSPS)

# write_csv(x = st_postal, file =
"US_HU12/US_HU12_stateIntersection_building.csv")

floodAreas2 <- dplyr::left_join(floodAreas, st_postal) %>%
dplyr::filter(!is.na(postal1))
floodAreas2 <- floodAreas2[!duplicated(floodAreas2$FIRST_huc1), ]

getOSM_buildings(floodPlain =floodAreas2, stateAbbrev = abbrev,
outputFolder = "outputs/osm/rerun/buildings" )
```

```
## RR
rr1 <- read_csv("outputs/osm/compiledOutputs/roads_rails.csv")
fp3 <- floodAreas[!floodAreas$FIRST_huc1 %in% rr1$FIRST_huc1, ]

# this takes a long time so I should write it out
s2 <- st_intersection(x = fp3, y = states)

fp4 <- fp3[!fp3$FIRST_huc1 %in% s2$FIRST_huc1, ]
write_csv(x = fp4, file = "US_HU12/US_HU12_noStates_rr.csv")

# Join results to flood area file
st_postal <- s2 %>%
  st_drop_geometry()%>%
  group_by(FIRST_huc1)%>%
  dplyr::select(FIRST_huc1,postal1 =  STUSPS)

write_csv(x = st_postal, file =
"US_HU12/US_HU12_stateIntersection_rr.csv")

floodAreas2 <- dplyr::left_join(floodAreas, st_postal)%>%
dplyr::filter(!is.na(postal1))
floodAreas2 <- floodAreas2[!duplicated(floodAreas2$FIRST_huc1), ]

getOSM_r_r(floodPlain =floodAreas2, stateAbbrev = abbrev, outputFolder =
"outputs/osm/rerun/roadsRails")


### combine reran content with existing material
# full files
b1 <- read_csv("outputs/osm/compiledOutputs/buildings.csv")
r1 <- read_csv("outputs/osm/compiledOutputs/roads_rails.csv")
# reruns
b2<- list.files(path = "outputs/osm/rerun/buildings", full.names = TRUE)
for(i in seq_along(b2)){
  b3 <- read_csv(b2[i])%>%
    dplyr::mutate("FIRST_huc1" = as.character(FIRST_huc1))
  if(i == 1){
    b4 <- b3
  }else{
    b4 <- bind_rows(b4, b3)
  }
}
write_csv(x = b4, file
="outputs/osm/rerun/buildings/compiled/buildingRerun.csv")

r2<- list.files(path = "outputs/osm/rerun/roadsRails", full.names = TRUE,
pattern = ".csv")
for(i in seq_along(r2)){
  r3 <- read_csv(r2[i])%>%
    dplyr::mutate("FIRST_huc1" = as.character(FIRST_huc1))
  if(i == 1){
    r4 <- r3
  }else{
    r4 <- bind_rows(r4, r3)
```

```
  }
}

write_csv(x = r4, file
="outputs/osm/rerun/roadsRails/compiled/roadsRailsRerun.csv")



# National Levee Database ------------------------------------------
leeves_all <- st_read("National Data Downloads/SP 22/NLD/nld.shp") %>% #
export features from geodatabase.
  st_transform(crs = st_crs(floodAreas))

# construct dataframe to hold all records
df <- data.frame(matrix(nrow = nrow(floodAreas), ncol = 3))
names(df) <- c("FIRST_huc1", "Leeve_area", "area")
df$area <- floodAreas$FIRST_area

time <- Sys.time()
nrow(floodAreas)
for(i in 1:nrow(floodAreas)){
  df$FIRST_huc1[i] <- floodAreas$FIRST_huc1[i]
  df$Leeve_area[i] <- getLeveeArea(floodPlain = floodAreas[i,],leeves =
leeves_all)
  # counter for progress
  if(i %% 10 == 0){
    print(paste0(i ," out of ", nrow(df)))
  }
  if(i %% 100 ==0){
    print(paste0((Sys.time() - time), " writing file"))
    write_csv(x = df[(i-100):i, ],
              file = paste0("outputs/leeveArea/leeveArea",i - 100,
"_",i,".csv")
    )
  }
  if(i == 82511){
    print(paste0((Sys.time() - time), " writing file"))
    write_csv(x = df[(i-11):i, ],
              file = paste0(outputFolder, "outputs/leeveArea/leeveArea",i
- 11, "_",i,".csv")
    )
  }
}
# compile outputs
compileOutputs(filePath = "outputs/levveArea/compiledLeeveArea",
               fileName = paste0("compiledLeeveArea_",Sys.Date()))
# ------------------------------------------------------------------------
-

# Global Forest Loss Dataset ---------------------------------------------
-
rasters <- list.files("National Data Downloads/SP 22/Forest Loss", pattern
= ".tif", full.names = TRUE)
```

```
rasters <- lapply(rasters[stringr::str_ends(rasters, pattern = ".tif")],
FUN = rast)

# construct dataframe to hold all records
df <- data.frame(matrix(nrow = nrow(floodAreas), ncol = 3))
names(df) <- c("FIRST_huc1", "Forest_Lost", "area")
df$area <- floodAreas$FIRST_area
time <- Sys.time()

for(i in 82501:nrow(floodAreas)){
  df$Forest_Lost[i] <- getForestLost(floodPlain = floodAreas[i,],
rasterList = rasters,multiple = FALSE)
  df$FIRST_huc1[i] <- floodAreas$FIRST_huc1[i]
  # counter for progress
  if(i %% 10 == 0){
    print(paste0(i ," out of ", nrow(df)))
  }
  if(i %% 100 ==0){
    print(paste0((Sys.time() - time), " writing file"))
    write_csv(x = df[(i-100):i, ],
              file = paste0("outputs/forestLost/forestLost",i - 100,
"_",i,".csv")
    )
  }
  if(i == 82511){
    print(paste0((Sys.time() - time), " writing file"))
    write_csv(x = df[(i-11):i, ],
              file = paste0("outputs/forestLost/forestLost",i - 11,
"_",i,".csv")
    )
  }
}
compileOutputs(filePath = "outputs/forestLost",
               fileName =
paste0("/compiledForestLost/compiledForestLost_",Sys.Date()))
### rerun features with multiple states
mStates <-
read_csv("outputs/forestLost/compiledForestLost/compiledForestLost_2022-
03-07.csv")%>%
  dplyr::filter(is.na(Forest_Lost))

mFlood <- floodAreas[floodAreas$FIRST_huc1 %in% mStates$FIRST_huc1, ]


### Rerun classification process
# construct dataframe to hold all records
df <- data.frame(matrix(nrow = nrow(mFlood), ncol = 3))
names(df) <- c("FIRST_huc1", "Forest_Lost", "area")
df$area <- mFlood$FIRST_area
time <- Sys.time()

for(i in 1:nrow(mFlood)){
  df$Forest_Lost[i] <- getForestLost(floodPlain = mFlood[i,], rasterList =
rasters,
```

```
                                      multiple = TRUE)
  df$FIRST_huc1[i] <- mFlood$FIRST_huc1[i]
  # counter for progress
  if(i %% 10 == 0){
    print(paste0(i ," out of ", nrow(df)))
  }
  if(i %% 100 ==0){
    print(paste0((Sys.time() - time), " writing file"))
    write_csv(x = df[(i-100):i, ],
              file =
paste0("outputs/forestLost/mutliple/forestLost_multiple",i - 100,
"_",i,".csv")
    )
  }
  if(i == nrow(mFlood)){
    print(paste0((Sys.time() - time), " writing file"))
    write_csv(x = df[(i-100):i, ],
              file =
paste0("outputs/forestLost/mutliple/forestLost_multiple",i - 100,
"_",i,".csv")
    )
  }
}
### replace values with previous NA
df <-
read_csv("outputs/forestLost/compiledForestLost/compiledForestLost_2022-
03-09.csv", col_types = cols(.default = "c"))

files <- list.files("outputs/forestLost/mutliple", pattern = ".csv",
full.names = TRUE)
for(i in seq_along(files)){
  f1 <- read_csv(files[i],progress = FALSE) %>%
    dplyr::mutate("FIRST_huc1" = as.character(FIRST_huc1))
  if(i == 1){
    df <- f1
  }else{
    df <- bind_rows(df, f1)
  }
}
## remove for dupicated features
lc2 <- df[!duplicated(df$FIRST_huc1), ]

## read in previous compiled data
allFeatures <-
read_csv("outputs/forestLost/compiledForestLost/compiledForestLost_2022-
03-09.csv")

allFeatures <- allFeatures[!allFeatures$FIRST_huc1 %in% lc2$FIRST_huc1, ]

df3 <- bind_rows(allFeatures,lc2)
write_csv(x= df3, file =
"outputs/forestLost/compiledForestLost/combinedWithMultiples_20220311.csv"
)
```

```
### evaluate the Current NA values
d1 <-
read_csv("outputs/forestLost/compiledForestLost/combinedWithMultiples_2022
0311.csv")
d2 <- d1[is.na(d1$Forest_Lost),]
View(d2)
# two features are not listed in numerical format
missingHucs <- floodAreas[!floodAreas$FIRST_huc1 %in% df3$FIRST_huc1,]
missingHucs2 <- c(missingHucs$FIRST_huc1, d2$FIRST_huc1[3:4])

# rerun content

fp2 <- floodAreas[floodAreas$FIRST_huc1 %in% missingHucs$FIRST_huc1, ]


df <- data.frame(matrix(nrow = nrow(fp2), ncol = 3))
names(df) <- c("FIRST_huc1", "Forest_Lost", "area")
df$area <- fp2$FIRST_area
time <- Sys.time()



for(i in 1:nrow(fp2)){
  df$Forest_Lost[i] <- getForestLost(floodPlain = fp2[i,], rasterList =
rasters,
                                     multiple = TRUE)
  df$FIRST_huc1[i] <- fp2$FIRST_huc1[i]
  # counter for progress
  if(i == nrow(fp2)){
    print(paste0((Sys.time() - time), " writing file"))
    write_csv(x = df[(i-nrow(fp2)):i, ],
              file =
paste0("outputs/forestLost/mutliple/forestLost_na_0311",i - nrow(fp2),
"_",i,".csv")
    )
  }
}
# append NA checks to the compiled data
d1 <-
read_csv("outputs/forestLost/compiledForestLost/combinedWithMultiples_2022
0311.csv")
d2 <- d1[!is.na(d1$Forest_Lost),]
d2a <- d1[is.na(d1$Forest_Lost),]
# drop
df1 <- df[df$FIRST_huc1 %in% d2a$FIRST_huc1, ]

df2 <- bind_rows(d2, df[2:3, ])
df2 <- df2[!duplicated(df2$FIRST_huc1), ]
write_csv(x = df2,
"outputs/forestLost/compiledForestLost/combinedWithMultiples_20220311_NAch
eck.csv")

# double check 131000000000 and rerun 100 features to test for systematic
errors
```

```
fp3 <- floodAreas[floodAreas$FIRST_huc1 ==  "131000000000", ]
val <- getForestLost(floodPlain = fp3, rasterList = rasters,
                                    multiple = FALSE)
# rerun content
fp4 <- floodAreas[1:100,]
df <- data.frame(matrix(nrow = nrow(fp4), ncol = 3))
names(df) <- c("FIRST_huc1", "Forest_Lost", "area")
df$area <- fp4$FIRST_area
time <- Sys.time()

for(i in 1:nrow(fp4)){
  df$Forest_Lost[i] <- getForestLost(floodPlain = fp4[i,], rasterList =
rasters,
                                     multiple = TRUE)
  df$FIRST_huc1[i] <- fp4$FIRST_huc1[i]
}



# ------------------------------------------------------------------------
-

# National Landcover Database ------------------------------------------
-
## land cover
nlcd_lc <-terra::rast("L:/Projects_active/EnviroScreen/data/NLCD/Land
Cover/nlcd_2019_land_cover_l48_20210604.img")
## render landcover
lc <- getNLCD_lc(polygons = floodAreas,landcover = nlcd_lc, outputFolder =
"F:/geoSpatialCentroid/floodplainIntagrety/outputs/nlcd_lc", startIndex =
69441)

## impervious
nlcd_impervious <- terra::rast("National Data Downloads/SP 22/NLCD/2019
percent imperviousness/nlcd_2019_impervious_l48_20210604.img")
## render impervious
getNLCD_impervious(polygons = floodAreas, impervious = nlcd_impervious,
                   outputFolder = "outputs/impervious", startIndex = 1)
compileOutputs(filePath = "outputs/impervious", fileName =
"compiled/impervious_20220301")
# ------------------------------------------------------------------------
-

# NHDPLUS  ---------------------------------------------------------------
-
### need to determine how to index features from the full element. should
be able to follow
### methods from the leeve dataset, with a length rather then area measure

canals <- st_read("National Data
Downloads/NHD_Canal_ditch/canal_floodplain_join.shp")
canalData <- getCanals(floodPlain = floodAreas , canals = canals)
write_csv(x = canalData, file = "outputs/canals/canals.csv")
```

```
# ------------------------------------------------------------------------
-

# Active Groundwater Level Network  ------------------------------------
-
wells <- st_read("National Data Downloads/SP 22/Groundwater
Wells/awl_wells/awl_wells.shp")
getGW_Wells(polygons = floodAreas,wells = wells, outputFolder
="outputs/wells", startIndex = 1)
compileOutputs(filePath = "outputs/wells", fileName =
"summary/wells_20220218")

### rerun the 511 that did not successfully run
errorWells <- read.csv("National Data Downloads/SP 22/Groundwater
Wells/missing wells.csv")
eFA <- floodAreas[floodAreas$FIRST_huc1 %in% errorWells$missing.HUC12, ]
getGW_Wells(polygons = eFA,wells = wells, outputFolder
="outputs/wells/rerun", startIndex = 1)


# add the rerun features back into the full compiled files
allWells <- read_csv("outputs/wells/summary/wells_20220218.csv")
reruns <- read_csv("outputs/wells/rerun/wells_0_511.csv")%>%
  dplyr::mutate(FIRST_huc1 = as.character(FIRST_huc1))
wells <- bind_rows(allWells, reruns)
write_csv(x = wells, file = "outputs/wells/summary/wells_20220309.csv")
# ------------------------------------------------------------------------
-

# Landfire Existing Vegetation type  -----------------------------------
-

landfire <- terra::rast("K:/Kira Simonson/National Data Downloads/SP
22/Landfire (vegetation type)/LF2016_EVT_200_CONUS/Tif/LC16_EVT_200.tif")
terra::activeCat(landfire)<-"EVT_GP"

#render vegetation types
veg <- getlandfire(polygons = floodAreas,EVT = landfire, outputFolder =
"K:/Kira Simonson/Stressor Densities/veg_csv", startIndex =71301)

##
###
# kira processing script for running content

# 2022-02-09
###

# install.packages("pacman")
pacman::p_load(sf,terra,dplyr, tictoc, readr)


# source scripts ---------------------------------------------------------
-
```

```
for(feature in list.files(path = "K:/Kira Simonson/src" ,full.names =
TRUE)){
  source(feature)
}


# Calculate Stressor Dataset Density ----------------------------------
-
## reading from a local file, taking a long time to load pulling from
floodAreas <- sf::st_read("K:/Kira Simonson/US_HU12/US_HU12.shp")

# Landfire Existing Vegetation type  ----------------------------------
-

landfire <- terra::rast("K:/Kira Simonson/National Data Downloads/SP
22/Landfire (vegetation type)/LF2016_EVT_200_CONUS/Tif/LC16_EVT_200.tif")
terra::activeCat(landfire)<-"EVT_GP"

veg <- getlandfire(polygons = floodAreas,EVT = landfire, outputFolder =
"K:/Kira Simonson/Stressor Densities/veg_csv", startIndex =71301)


# It's unlikely this will ever return an object due to run time
# write_csv(veg, paste0("outputs/landfire",Sys.Date(),".csv"))

# alternative : compile and render from written files
files <- list.files("K:/Kira Simonson/Stressor Densities/R_CSV_Outputs",
pattern = ".csv", full.names = TRUE)
for(i in seq_along(files)){
  f1 <- read_csv(files[i])
  # feature 226; id column was write as double not charater.
  f1$FIRST_huc1 <- as.character(f1$FIRST_huc1)
  if(i == 1){
    veg <- f1
  }else{
    veg <- bind_rows(veg, f1)
  }
}
### some duplicated features
veg2 <- veg[!duplicated(veg$FIRST_huc1), ]

## some features meeting
ids <- floodAreas$FIRST_huc1[!as.character(floodAreas$FIRST_huc1) %in%
veg2$FIRST_huc1]
## rerun these features
fl2 <- floodAreas[floodAreas$FIRST_huc1 %in% ids, ]
veg <- getlandfire(polygons = floodAreas,EVT = landfire, outputFolder =
"K:/Kira Simonson/Stressor Densities/R_CSV_Outputs", startIndex = 1)
veg2 <- bind_rows(veg2, veg)
veg3 <- veg2[!duplicated(veg2$FIRST_huc1), ]
### still a bit highed then input features
veg3_e <- veg3[veg3$FIRST_huc1 %in% floodAreas$FIRST_huc1, ]
# errors were from features with Huc id save in scitentic notation.
```

```
write_csv(veg3_e, paste0("K:/Kira Simonson/Stressor
Densities/R_CSV_Outputs",Sys.Date(),".csv"))



##rstudio for floodplain, NHD, and HA metrics
library(tidyverse)
library(terra)
library(rgdal)
library(tidyr)
library(broom)
library(raster)


onedrive <- "C:\\Users\\kmsimon\\OneDrive - Colostate\\Floodplain
Integrity Research"
output <-"K:\\Kira Simonson\\Stressor Densities\\compiled outputs\\"

setwd(onedrive)


# need to intersect NHD flow lines with HUC12 floodplain units - will need
shapefiles for both
NHDfloodplains<-sf::st_read("K:\\Kira
Simonson\\Data_from_Katie\\nhd_floodplain_join.shp")

##this is what i need to create, HUC12 shapefile with associated NHD
floodlines

#NHD flow lines with HUC12 floodplain units-  convert shapefile attribute
tables to data frame and tibble
US_HUC12_NHD_df <- as.data.frame(NHDfloodplains)
US_HUC12_NHD_tibble<- as_tibble(US_HUC12_NHD_df)


#get HA metrics
US_HA_1 <- read_delim("C:\\Users\\kmsimon\\OneDrive -
Colostate\\Floodplain Integrity Research\\R Code HA Metrics\\Predicted
Hydrologic Alteration Values\\Pred_US_1.txt")
US_HA_2 <- read_delim("C:\\Users\\kmsimon\\OneDrive -
Colostate\\Floodplain Integrity Research\\R Code HA Metrics\\Predicted
Hydrologic Alteration Values\\Pred_US_2.txt")
# convert to data frame
US_HA_1_df <- data.frame(US_HA_1)
US_HA_2_df <- data.frame(US_HA_2)
# convert to tibble
US_HA_1 <- as_tibble(US_HA_1_df)
US_HA_2 <- as_tibble(US_HA_2_df)
#combing rows of the two data tables
US_HA <- rbind(US_HA_1,US_HA_2)
US_HA <- as_tibble(US_HA)
```

```
# join floodplain units to HA metric by COMID
US_FP_HA = US_HUC12_NHD_tibble %>% left_join(US_HA, by = 'COMID')


# take the max stream order mean values for HA
US_HA_Mean <-  US_FP_HA %>% group_by(FIRST_1,FIRST_n) %>%
summarise(streamorder = mean(mx_StrO,na.rm = TRUE),MH20avg =
mean(pnMH20,na.rm = TRUE))
US_HA_Mean$FIRST_1 <- as.numeric(US_HA_Mean$FIRST_1)

#rename columns
names(US_HA_Mean)[names(US_HA_Mean) == "FIRST_1"] <- "HUC12"
names(US_HA_Mean)[names(US_HA_Mean) == "FIRST_n"] <- "HUC12_name"

#new FP info

FP_info <- read.csv("K:\\Kira Simonson\\Stressor Densities\\Stressor
Data\\FP_info1.csv")
try <-merge(FP_info,US_HA_Mean, by.x = "FIRST_huc1",by.y="HUC12")

write_csv(US_HA_Mean,file = paste0(output,"HA_compiled.csv"))

library(tidyverse)
library(sf)

sf_use_s2(FALSE)

US_HU12 <- st_read("K:\\Kira Simonson\\Data_from_Katie\\floodplains.shp")
%>% st_transform(4269) %>% st_zm()

nhd <- st_read("K:\\Kira Simonson\\Data_from_Katie\\nhd.shp") %>% st_zm()

nhd_tabular <- read_csv("K:\\Kira
Simonson\\Data_from_Katie\\nhd_tabular.csv") %>%
select(COMID,StreamOrde,FTYPE)

# table of all floodplain features linked to highest nhd features (only
highest stream orders per floodplain unit)
# floodplain2nhd <- st_join(US_HU12, nhd, join = st_intersects, left =
FALSE) %>%
#   left_join(., nhd_tabular, by = "COMID") %>%
#   filter(FTYPE == "StreamRiver" | FTYPE == "ArtificialPath") %>%
#   group_by(FIRST_huc1) %>%
#   mutate(max_StreamOrde = max(StreamOrde)) %>%
#   ungroup() %>%
#   filter(StreamOrde == max_StreamOrde) %>%
#   distinct(FIRST_huc1, StreamOrde, .keep_all = TRUE) %>%
#
select(FIRST_huc1,FIRST_name,COMID,FTYPE,StreamOrde,max_StreamOrde,geometr
y)
# st_write(floodplain2nhd, 'floodplain_nhd_join.shp')
```

```
#shapefile of nhd features linked to floodplain units (only highest stream
orders per floodplain unit)
nhd2floodplain <- st_join(nhd, US_HU12, join = st_intersects, left =
FALSE) %>%
  left_join(., nhd_tabular, by = "COMID") %>%
  filter(FTYPE == "StreamRiver" | FTYPE == "ArtificialPath") %>%
  group_by(FIRST_huc1) %>%
  mutate(max_StreamOrde = max(StreamOrde)) %>%
  ungroup() %>%
  filter(StreamOrde == max_StreamOrde) %>%
  distinct(FIRST_huc1, StreamOrde, .keep_all = TRUE) %>%

select(FIRST_huc1,FIRST_name,COMID,FTYPE,StreamOrde,max_StreamOrde,geometr
y)
st_write(nhd2floodplain, 'nhd_floodplain_join.shp')



### creating CanalDitch shapefile ------------------------------
canals2floodplain <- st_join(nhd, US_HU12, join = st_intersects, left =
FALSE) %>%
  left_join(., nhd_tabular, by = "COMID") %>%
  filter(FTYPE == "CanalDitch")
st_write(canals2floodplain, paste0("K:\\Kira Simonson\\National Data
Downloads\\NHD_Canal_ditch", '\\canal_floodplain_join.shp'))


#' getNLCD_lc
#'
#' @param polygons : flood plain area feature as an sf object
#' @param landcover : nlcd as a terra rast object
#' @param outputFolder : location where intermediate steps will be saved
#' @param startIndex : where in the feature generation should the process
start. Implimented because getting a full run is unlikley.
#'

getNLCD_lc <- function(polygons,landcover, outputFolder, startIndex){
  # https://www.mrlc.gov/data/legends/national-land-cover-database-2019-
nlcd2019-legend
  l1 <- c(22:24,81:82) # development low, medium, high intensity and
pasture/hay or cultivated crops

  # construct dataframe to hold all records
  df <- data.frame(matrix(nrow = nrow(polygons), ncol = 3 + length(l1)))
  names(df) <- c("FIRST_huc1", l1, "total", "area")
  df$area <- polygons$FIRST_area

  time <- Sys.time()
  # run through each polygon
  for(i in startIndex:length(polygons$FIRST_huc1)){
    # select feature
    t1 <- vect(polygons[i,])
    # assgin Id
    df$FIRST_huc1[i] <- t1$FIRST_huc1  #again change when we have data
```

```
    # crop and mask image
    r2 <- landcover %>% terra::crop(t1)%>%terra::mask(t1)

    # determine values
    vals <- values(r2)
    #drop NA and 0
    vals[vals == 0] <- NA
    vals <- vals[!is.na(vals)]
    # total number of land based cells
    total <- length(vals)
    df[i,"total"] <- total
    # unique land cover class in area
    uniqueVals <- sort(unique(vals))
    for(j in seq_along(l1)){
      code <- as.numeric(l1[j])
      # test if the specific land cover class is present in subset area
      if(code %in% uniqueVals){
        index <- code
        val2 <- vals[vals == index ]
        sum1 <- length(val2)
        percentCover <- (sum1/total)*100
        df[i,as.character(index)] <- percentCover
      }
    }
    # counter for progress
    if(i %% 10 == 0){
      print(paste0(i ," out of ", nrow(df)))
    }
    if(i %% 100 ==0){
      print(paste0((Sys.time() - time), " writing file"))
      write_csv(x = df[(i-100):i, ],
                file = paste0(outputFolder, "/nlcd_lc_",i - 100,
"_",i,".csv")
                )
    }
    if(i == 82511){
      print(paste0((Sys.time() - time), " writing file"))
      write_csv(x = df[(i-11):i, ],
                file = paste0(outputFolder, "/nlcd_lc",i - 11,
"_",i,".csv")
      )
    }
  }
}

#' getLandfire
#'
#' @param polygons : flood plain area feature as an sf object
#' @param EVT : landfire existing vegetation type as a terra rast object
#' @param outputFolder : location where intermediate steps will be saved
#' @param startIndex : where in the feature generation should the process
start. Implemented because getting a full run is unlikely.
#'
```

```
#' @return dataframe of percent area for landfire existing vegetation
type,EVT, Groups 701-709,731 classes for all flood plain areas.
#'

getlandfire <- function(polygons,EVT,outputFolder, startIndex){
  # https://landfire.gov/evt.php
  l1 <-
c(9327,9301,9302,9307,9308,9309,9310,9311,9316,9317,9318,9319,9320,9321

,9323,9324,9325,9326,9328,9329,9332,9336,9337,9810,9811,9816,9817,9823
          ,9825,9826,9827,9828,9829,9312,9322) # which represent various
types of invasive, non-agricultural plant species

  # construct dataframe to hold all records
  df <- data.frame(matrix(nrow = nrow(polygons), ncol = 3 + length(l1)))
  names(df) <- c("FIRST_huc1", l1, "total", "area")
  df$area <- polygons$FIRST_area

  time <- Sys.time()
  # run through each polygon
  for(i in startIndex:length(polygons$FIRST_huc1)){

    try({
    # select feature
    t1 <- vect(polygons[i,])
    # assgin Id
    df$FIRST_huc1[i] <- t1$FIRST_huc1  #again change when we have data
    # crop and mask image
    r2 <- EVT %>% terra::crop(t1)%>%terra::mask(t1)

    # determine values
    vals <- values(r2)
    #drop NA and 0
    vals[vals == 0] <- NA
    vals <- vals[!is.na(vals)]
    # total number of land based cells
    total <- length(vals)
    df[i,"total"] <- total
    # unique land cover class in area
    uniqueVals <- sort(unique(vals))
    for(j in seq_along(l1)){
      code <- as.numeric(l1[j])
      # test if the specific land cover class is present in subset area
      if(code %in% uniqueVals){
        index <- code
        val2 <- vals[vals == index ]
        sum1 <- length(val2)
        percentCover <- (sum1/total)*100
        df[i,as.character(index)] <- percentCover
      }
    }
    })
    # counter for progress
    if(i %% 10 == 0){
```

```
      print(paste0(i ," out of ", nrow(df)))
    }
    if(i %% 100 ==0){
      print(paste0((Sys.time() - time), " writing file"))
      write_csv(x = df[(i-100):i, ],
                file = paste0(outputFolder, "/landfire_",i - 100,
"_",i,".csv")
      )
    }
    if(i == 82511){
      print(paste0((Sys.time() - time), " writing file"))
      write_csv(x = df[(i-11):i, ],
                file = paste0(outputFolder, "/landfire_",i - 11,
"_",i,".csv")
      )
    }


    }
  return(df)
}

# trying to use the purrr map function. We want to test
getForestLost <- function(floodPlain, rasterList, multiple){
  # test for intersection between features
  f1 <- floodPlain %>%
    st_transform(crs = terra::crs(rasterList[[1]]))
  bbox <- st_bbox(f1)

  if(isTRUE(multiple)){
    r1 <- NA
    for(i in seq_along(rasterList)){
      r2 <- terra::ext(rasterList[i][[1]])
      # condition for testing overlap between features
      r3 <- rasterList[i][[1]]
      t <- NA
      try(t <- crop(r3,f1))
      if(class(t) == "SpatRaster"){
        if(class(r1) != "SpatRaster"){
          r1 <- t
        }else{
          r1 <- terra::merge(r1,t)
        }
      }
    }
    r1 <- r1 %>% terra::mask(vect(f1))
    # all values
    v1 <- values(r1)
    vals <- v1[!is.na(v1)]
    total <- length(vals)
    # assuming negitive implies forest loss
    decrease <- length(vals[vals > 0])
    # calculate percent
    if(isTRUE(total == 0)){
```

```
      value <- 0
    }else{
      value <- (decrease/total)*100

    }
    }else{
    for(i in seq_along(rasterList)){
      r2 <- terra::ext(rasterList[i][[1]])
      # condition for testing overlap between features
      if( r2[1] < bbox[1] & r2[2] > bbox[3] & r2[3] < bbox[2] & r2[4] >
bbox[4]){
        r1 <- rasterList[i][[1]]
        print(i)
        # percentage of cells that reported forest lost
        fp2 <- vect(f1)
        # sort to extent of the feature of interest
        r3 <- r1 %>%
          terra::crop(fp2)%>%
          terra::mask(fp2)
        # all values
        v1 <- values(r3)
        vals <- v1[!is.na(v1)]
        total <- length(vals)
        # assuming negitive implies forest loss
        decrease <- length(vals[vals > 0])
        # calculate percent
        if(isTRUE(total == 0)){
          value <- 0
        }else{
          value <- (decrease/total)*100

        }
        break
      }else{
        # this should capture all features that overlap between
boundaries. These will have to be
        # calculated elsewhere
        value <- NA
      }
    }
  }

  return(value)
}

#' get impervious surface
#' @description : generate the average percent impervious surface for all
features in the floodplain
#' @param polygons : floodplain polygon features
#' @param impervious : NLCD 2019 impervious surface layer as a terra::rast
#' @param outputFolder : location for storing outputs
#' @param startIndex : where to start the itorative process from. numeric
value betwee 1 and nrow(floodplain)
```

```
getNLCD_impervious <- function(polygons,impervious, outputFolder,
startIndex){
  # https://www.mrlc.gov/data?f%5B0%5D=category%3Aurban%20imperviousness
  # raster values store % of area considered impervious.
  # Percent imperviousness (average percent imperviousness values reported
for each 30 m cell for all cells in the floodplain)

  # construct dataframe to hold all records
  df <- data.frame(matrix(nrow = nrow(polygons), ncol = 3))
  names(df) <- c("FIRST_huc1", "aveImpervious", "area")
  df$area <- polygons$FIRST_area

  time <- Sys.time()
  # run through each polygon
  for(i in startIndex:length(polygons$FIRST_huc1)){
    # select feature
    t1 <- vect(polygons[i,])
    # assign Id
    df$FIRST_huc1[i] <- t1$FIRST_huc1  #again change when we have data
    # crop and mask image
    try(
      # calculate the average impervious values across all features in
floodplain
      r1 <- impervious %>%
        terra::crop(t1)%>%
        terra::mask(t1),
    )
    if(class(r1)=="SpatRaster"){
      vals <- values(r1)
      vals[vals == 127] <- NA
      df$aveImpervious[i] <- vals %>% na.omit() %>% mean()
    }else{
      df$aveImpervious[i] <- NA
    }
    rm(r1)


    # counter for progress
    if(i %% 10 == 0){
      print(paste0(i ," out of ", nrow(df)))
    }
    if(i %% 100 ==0){
      print(paste0((Sys.time() - time), " writing file"))
      write_csv(x = df[(i-100):i, ],
                file = paste0(outputFolder, "/impervious_",i - 100,
"_",i,".csv")
      )
    }
    if(i == 82511){
      print(paste0((Sys.time() - time), " writing file"))
      write_csv(x = df[(i-11):i, ],
                file = paste0(outputFolder, "/impervious_",i - 11,
"_",i,".csv")
      )
```

```
    }
  }
}



getLeveeArea <- function(floodPlain, leeves){

  # calculate the area of leeve features

  ## this might be slow... but it should
  crop <- st_crop(x = leeves_all, y = floodPlain)
  if(nrow(crop) >0){# some test to see if any features are in th ecrop
area
    # generate the overlapping areas between the features
    intersection <- st_intersection(crop, floodPlain)
    # calculate area
    if(nrow(intersection) >0 ){
      area <- st_area(intersection)*0.000001
    }else{
      area <- 0
    }
  }else{
    area <- 0
  }
  return(area)
}



getGW_Wells <- function(polygons, wells, outputFolder, startIndex){

  # construct dataframe to hold all records
  df <- data.frame(matrix(nrow = nrow(polygons), ncol = 3))
  names(df) <- c("FIRST_huc1", "numberWells", "area")
  df$area <- polygons$FIRST_area


  # reproject wells
  wells <- wells %>%
    st_transform(crs =  st_crs(floodAreas))
  ### mutate was occurring before transformation Split out to dely
  w1 <- wells %>%
    # we want lat lon values as rows in df to filter on.
    mutate(lon = unlist(map(wells$geometry,1)),
           lat = unlist(map(wells$geometry,2)))

  # rather then spatial operation use the boundary box to filter wells
  # than test for intersection
  time <- Sys.time()
  for(i in startIndex:nrow(polygons)){
    ## generate count of well per floodplain area
    # subset flood area
    t1 <- polygons[i, ]
```

```
    df$FIRST_huc1[i] <- t1$FIRST_huc1

    # get square area containing flood plain for first filter
    bb <- sf::st_bbox(t1)
    # filter wells based on bbox
    w2 <- w1 %>%
      dplyr::filter(lon >= bb[1], lon <= bb[3], lat >= bb[2], lat <=
bb[4])
    # test for presences on tabular filter
    if(nrow(w2) == 0){
      df$numberWells[i] <- 0
    }else{
      # test for intersection between well and floodplain
      inter <- st_intersects(x = t1, y = w2) %>% unlist()
      if(length(inter)== 0){
        # Not intersecting
        df$numberWells[i] <- 0
      }else{
        # number of intersecting wells
        df$numberWells[i] <- length(inter)
      }
    }
    # counter for progress
    if(i %% 100 == 0){
      print(paste0(i ," out of ", nrow(df)))
    }
    # write every 1000 iterations
    if(i %% 1000 ==0){
      print(paste0((Sys.time() - time), " writing file"))
      write_csv(x = df[(i-1000):i, ],
                file = paste0(outputFolder, "/wells_",i - 1000,
"_",i,".csv")
      )
    }
    # write every 1000 iterations
    if(i == nrow(df)){
      print(paste0((Sys.time() - time), " writing file"))
      write_csv(x = df[(i-nrow(df)):i, ],
                file = paste0(outputFolder, "/wells_",i - nrow(df),
"_",i,".csv")
      )
    }
  }
}



getCanals <- function(floodPlain, canals){
  # input dataset has already been joined to the flood plains
  # need to reproject and recalculate lengths in measured units
  df <- canals %>%
    dplyr::select(FIRST_huc1)%>%
    sf::st_transform(crs = st_crs(floodPlain))%>%
    dplyr::mutate(
```

```
      # length is in meters so translate to km
      length = as.numeric(sf::st_length(canals)*0.001)
    )%>%
    sf::st_drop_geometry()%>%
    group_by(FIRST_huc1) %>%
    summarise(totalLength = sum(length), totalDitches = n())
  # join this floodplain featues and assign true zero values
  fp2 <- floodPlain %>%
    st_drop_geometry()%>%
    dplyr::left_join(df)%>%
    dplyr::mutate(
      totalLength =
        dplyr::case_when(
          is.na(totalLength) ~ 0,
          TRUE ~ totalLength
        )
    )%>%
    dplyr::select(FIRST_huc1, totalLength, totalDitches)

  return(fp2)

}
#' Compile OSM data
#'
#' @param buildings: TRUE/FALSE for if you are compiling the building or
road and rail layers
#' @param path : folder location for all OSM outputs
#'
#' @return : df of compiled and summaries input across all states

compileOSM <- function(buildings, path){

  if(isTRUE(buildings)){
    loc <- paste0(path, "/buildings")
    }else{
      loc <- paste0(path, "/roadsRails")
  }
  files <- list.files(loc, pattern = ".csv", full.names = TRUE)
  # loop over state
  for(i in seq_along(abbrev)){
    #grab all state features
    f2 <- files[grepl(pattern = abbrev[i],x = files, ignore.case = FALSE)]
    for(j in seq_along(f2)){
      f1 <- read_csv(f2[j]) %>%
        dplyr::mutate("FIRST_huc1" = as.character(FIRST_huc1))
      if(j == 1){
        df <- f1
      }else{
        df <- bind_rows(df, f1)
      }
    }
    # remove any duplicated features
    lc2 <- df[!duplicated(df$FIRST_huc1), ]
    if(i == 1){
```

```
      df2 <- df
    }else{
      df2 <- bind_rows(df2,df)
    }
  }
  if(isTRUE(buildings)){
    df3 <- df2 %>%
      group_by(FIRST_huc1)%>%
      dplyr::summarise(
        buildingArea = sum(buildingArea),
        count = n())
    }else{
      df3 <- df2 %>%
        group_by(FIRST_huc1)%>%
        dplyr::summarise(
          roadLength = sum(roadLength),
          railLength = sum(railLength ),
          totalLength = sum(totalLength),
          count = n())
    }
  return(df3)
}


getOSM_r_r <- function(floodPlain, stateAbbrev,outputFolder){
  # for each state
  abbrev <- stateAbbrev

  for(i in abbrev){
    features <- floodPlain[grep(pattern =i, x = floodPlain$postal1), ]
    # grab state
    st2 <- i
    #
    roads <- st_read(paste0("National Data Downloads/SP 22/Open Street
Map/US Roads/", st2,"_roads.shp"))%>%
      st_transform(crs = st_crs(features))
    rails <- st_read(paste0("National Data Downloads/SP 22/Open Street
Map/US Railroads/", st2,"_rail.shp"))%>%
      st_transform(crs = st_crs(features))


    # create a dataframe for outputs
    # construct dataframe to hold all records
    df <- data.frame(matrix(nrow = nrow(features), ncol = 5))
    names(df) <- c("FIRST_huc1", "roadLength", "railLength",
"totalLength", "area")
    df$area <- features$FIRST_area

    time <- Sys.time()
    # iterate over features
    for(j in seq_along(features$FIRST_huc1)){
      # select features of interest
      f1 <- features[j, ]
```

```
      df$FIRST_huc1[j] <- f1$FIRST_huc1

      # crop each element
      road1 <- sf::st_crop(roads, st_bbox(f1))
      rails1 <- sf::st_crop(rails, st_bbox(f1))
      print(paste0(j, ": of ", nrow(features), " for ", st2))

      # test for area of roads
      if(nrow(road1)>0){
        r2 <- st_intersection(x = road1, y = f1)
        # test for intersection
        if(nrow(r2) > 0 ){
          val <- st_length(r2) %>% sum()
          # save to dataframe and convert to kmsq
          df$roadLength[j] <- val*0.001
        }else{
          df$roadLength[j] <- 0
        }
      }else{
        df$roadLength[j] <- 0
      }
      # test for length of railroads
      if(nrow(rails1)>0){
        rl2 <- st_intersection(x = rails1, y = f1)
        # test for intersection
        if(nrow(rl2) > 0 ){
          val <- st_length(rl2) %>% sum()
          # save to dataframe and convert to kmsq
          df$railLength[j] <- val*0.001
        }else{
          df$railLength[j] <- 0
        }
      }else{
        df$railLength[j] <- 0
      }
      df$totalLength[j] <- sum(df$railLength[j],df$roadLength[j])
      # counter for progress
      if(j %% 10 == 0){
        #print(paste0(i ," out of ", nrow(df)))
      }
      if(j %% 100 ==0){
        print(paste0((Sys.time() - time), " writing file"))
        write_csv(x = df[(j-100):j, ],
                  file = paste0(outputFolder, "/",i,"_osm_r_r_",j - 100,
"_",j,".csv")
        )
      }
      if(j == nrow(df)){
        print(paste0((Sys.time() - time), " writing file"))
        write_csv(x = df[(j- nrow(df)):j, ], # this will create some
overlap but we can remove at compile step
                  file = paste0(outputFolder, "/",i,"_osm_r_r_",j -
nrow(df), "_",j,".csv"))
```

```
      }
    }

  }
}


getOSM_buildings <- function(floodPlain, stateAbbrev,outputFolder){
  # for each state
  abbrev <- stateAbbrev

    for(i in abbrev){
    features <- floodPlain[grep(pattern =i, x = floodPlain$postal1), ]
    # grab state
    st2 <- i
    #
    building <- st_read(paste0("National Data Downloads/SP 22/Open Street
Map/US Buildings/", st2,"_buildings.shp"))%>%
      st_transform(crs = st_crs(features))

    # create a dataframe for outputs
    # construct dataframe to hold all records
    df <- data.frame(matrix(nrow = nrow(features), ncol = 3))
    names(df) <- c("FIRST_huc1", "buildingArea", "area")
    df$area <- features$FIRST_area

    time <- Sys.time()
    # iterate over features
    for(j in seq_along(features$FIRST_huc1)){
      # select features of interest
      f1 <- features[j, ]
      df$FIRST_huc1[j] <- f1$FIRST_huc1
      # crop each element
      b1 <- sf::st_crop(building, st_bbox(f1))

      print(paste0(j, ": of ", nrow(df), " for ", st2))
      # test for area of buildings
      if(nrow(b1)>0){
        b2 <- st_intersection(x = b1, y = f1)
        # test for intersection
        if(length(b2[[1]]) > 0 ){
          print(f1$FIRST_huc1)
          val <- st_area(b2) %>% sum()
          # save to dataframe and convert to kmsq
          df$buildingArea[j] <- val*0.000001
        }else{
          df$buildingArea[j] <- 0
        }
      }else{
        df$buildingArea[j] <- 0
      }
      # counter for progress
      if(j %% 10 == 0){
        print(paste0(i ," out of ", nrow(df)))
```

```
      }
      if(j %% 100 ==0){
        print(paste0((Sys.time() - time), " writing file"))
        write_csv(x = df[(j-100):j, ],
                  file = paste0(outputFolder, "/",i,"_osm_building_",j -
100, "_",j,".csv")
        )
      }
      if(j == nrow(df)){
        print(paste0((Sys.time() - time), " writing file"))
          write_csv(x = df[(j- nrow(df)):j, ], # this will create some
overlap but we can remove at compile step
                    file = paste0(outputFolder, "/",i,"_osm_building_",j -
nrow(df), "_",j,".csv"))

      }
    }
  }
}
#MRB_main

#get script
for(feature in list.files(path = "K:\\Kira Simonson\\Stressor
Densities\\MRB\\MRB_Code\\Density Processing\\",full.names = TRUE)){
  print(feature)
  source(feature)
}

#output folder
MRBoutput <-"K:\\Kira Simonson\\Stressor Densities\\MRB\\MRBoutputs"


#get floodplains shapefile
MRBfloodplains <- sf::st_read("K:\\Kira Simonson\\Stressor
Densities\\MRB\\MRB Shapefile\\MRBfloodplains.shp")

#define landcover and YEAR

MRBlc<-"K:\\Kira Simonson\\National Data Downloads\\SP 22\\Mississippi
River Basin\\Input Data\\02_MRB_floodplain_LU"

n = 1941
#get landcover
landcover= (terra::rast(paste0(MRBlc,"\\LULC_FP_MRB",n,".tif")))
landcover[is.na(landcover)] <- 9
landcover <- project(landcover,"epsg:6350")


MRB <- getNLCD_MRB(polygons = MRBfloodplains,landcover,
outputFolder=MRBoutput, startIndex=1,n=n)


for (i in 0:11){
  n = 1945+5*i
```

```
  #get landcover
  landcover= (terra::rast(paste0(MRBlc,"\\LULC_FP_MRB",n,".tif")))
  landcover <- project(landcover,"epsg:6350")
  landcover[is.na(landcover)] <- 9

MRB <- getNLCD_MRB(polygons = MRBfloodplains,landcover,
outputFolder=MRBoutput, startIndex=1,n=n)}

## MRB Floodplain delineation and NLCD processing
# kira simonson spring 2022

# install.packages("pacman")
pacman::p_load(sf,terra,dplyr, readr,tools,data.table)



#' getNLCD_lc_MRB
#'
#' @param polygons : flood plain area feature as an sf object
#' @param landcover : nlcd as a terra rast object
#' @param outputFolder : location where intermediate steps will be saved
#' @param startIndex : where in the feature generation should the process
start. Implemented because getting a full run is unlikely.

getNLCD_MRB <- function(polygons,landcover, outputFolder, startIndex,n){


  l1 <- c(2,6) # development and agriculture

  # construct dataframe to hold all records
  df <- data.frame(matrix(nrow = nrow(polygons), ncol = 3 + length(l1)))
  names(df) <- c("FIRST_huc1", l1, "total", "area")
  df$area <- polygons$FIRST_area

  time <- Sys.time()
  # run through each polygon
  for(i in startIndex:length(polygons$FIRST_huc1)){
    # select feature
    t1 <- vect(polygons[i,])
    # assgin Id
    df$FIRST_huc1[i] <- t1$FIRST_huc1  #again change when we have data
    # crop and mask image
    r2 <- landcover %>% terra::crop(t1)%>%terra::mask(t1)


    # determine values
    vals <- values(r2)
    #drop NA and 0
    vals[vals == 0] <- NA
    vals <- vals[!is.na(vals)]
    # total number of land based cells
    total <- length(vals)
    df[i,"total"] <- total
    # unique land cover class in area
```

```
    uniqueVals <- sort(unique(vals))
    for(j in seq_along(l1)){
      code <- as.numeric(l1[j])
      # test if the specific land cover class is present in subset area
      if(code %in% uniqueVals){
        index <- code
        val2 <- vals[vals == index ]
        sum1 <- length(val2)
        percentCover <- (sum1/total)*100
        df[i,as.character(index)] <- percentCover
      }
    }
    # counter for progress
    if(i %% 10 == 0){
      print(paste0(i ," out of ", nrow(df)))
    }
    if(i %% 100 ==0){
      print(paste0((Sys.time() - time), " writing file"))
      write_csv(x = df[(i-100):i, ],
                file = paste0(outputFolder, "/MRB_",n,"_",i - 100,
"_",i,".csv")
      )
    }
    if(i == 34421){
      print(paste0((Sys.time() - time), " writing file"))
      write_csv(x = df[(i-21):i, ],
                file = paste0(outputFolder, "/MRB_",n,"_",i - 21,
"_",i,".csv")
      )
    }
  }
}
```

IFI calculation

The following R scripts were used to calculate functional and overall IFI values:

```
### stressor merging into singular dataframes for overall FP and for MRB
### kira simonson spring 2022


# install.packages("pacman")
pacman::p_load(dplyr,
readr,tools,data.table,tidyr,lsmeans,corrplot,foreign,RColorBrewer,ggcorrp
lot)


## overall FP ---------------------------------------------
basepath <- "K:\\Kira Simonson\\Stressor Densities"


##merging files into one data frame for overall FP------

# Load all csv files in folder into list
data.path <- paste(basepath, "\\Stressor Data\\", sep="")
filelist <- list.files(path = data.path, pattern="*.csv")


# read in each .csv file in folder and create a data frame with the same
name as the .csv file
for (i in 1:length(filelist)){
  assign(file_path_sans_ext(filelist[i]),
         read.csv(paste(data.path, filelist[i], sep='')) )}

## Get all information relevant into one df using FP HUC-12 as basis

#NLCD Ag
data.merge <- merge(FP_info1[,c("FIRST_huc1", "FIRST_area",
"mx_StrO","FTYPE")], AgArea[, c("HUC12", "Density")],by.x = "FIRST_huc1",
by.y="HUC12", all.x = TRUE)
setnames(data.merge,old=c("FIRST_huc1","FIRST_area","FTYPE","Density")
,new=c("HUC12", "HUC12_Areakm2","pathtype","Agriculture"))

#Buildings
data.merge <- merge(data.merge, Buildings[, c("HUC12", "area")], by =
"HUC12", all.x = TRUE)
colnames(data.merge)[colnames(data.merge)=="area"] <- "Buildings"


#Canal Ditch
data.merge <- merge(data.merge, CanalDitch[, c("HUC12", "totalLength")],
by = "HUC12", all.x = TRUE)
colnames(data.merge)[colnames(data.merge)=="totalLength"] <- "Ditches"

#NLCD Developed
```

```
data.merge <- merge(data.merge, DevelopedArea[, c("HUC12", "Density")], by
= "HUC12", all.x = TRUE)
colnames(data.merge)[colnames(data.merge)=="Density"] <- "Developed"

#Forest Loss
data.merge <- merge(data.merge, ForestLoss[, c("HUC12", "Loss")], by =
"HUC12", all.x = TRUE)
colnames(data.merge)[colnames(data.merge)=="Loss"] <- "ForestLoss"

#HA
data.merge <- merge(data.merge, HA[, c("HUC12", "MH20avg")], by = "HUC12",
all.x = TRUE)
colnames(data.merge)[colnames(data.merge)=="MH20avg"] <- "HA"

#Percent Impervious
data.merge <- merge(data.merge, Impervious[, c("HUC12", "Density")], by =
"HUC12", all.x = TRUE)
colnames(data.merge)[colnames(data.merge)=="Density"] <- "Impervious"


#Levee
data.merge <- merge(data.merge, LeveedArea[, c("HUC12", "Density")], by =
"HUC12", all.x = TRUE)
colnames(data.merge)[colnames(data.merge)=="Density"] <- "LeveedArea"


#Vegetation
data.merge <- merge(data.merge, Vegetation[, c("HUC12", "Density")], by =
"HUC12", all.x = TRUE)
colnames(data.merge)[colnames(data.merge)=="Density"] <- "NonNativeVeg"


#Roads Railroads
  data.merge <- merge(data.merge, RoadsRailroads[, c("HUC12",
"totalLength")], by = "HUC12", all.x = TRUE)
  colnames(data.merge)[colnames(data.merge)=="totalLength"] <-
"Roads_Rail"


#Wells
  data.merge <- merge(data.merge, Wells[, c("HUC12", "Count")], by =
"HUC12", all.x = TRUE)
  colnames(data.merge)[colnames(data.merge)=="Count"] <- "Wells"


data.merge <- data.merge %>% mutate_at(vars(-("pathtype")), function(x)
as.numeric(as.character(x)))

## Version with zeros changed to NA
data.merge.NA <- data.merge
data.merge.NA[data.merge.NA==0] <- NA
# version with NA changed to zero
data.merge[is.na(data.merge)] <- 0
```

```
# Save as .csv file
out.path <- paste(basepath, "\\MergedOutputs\\", sep="")
out.file <- paste(out.path, "Combined_Data.csv", sep="")
write.csv(data.merge, file = out.file, row.names = FALSE)

#
#

## MRB FP ------------------------------------------------------------
basepathMRB <- "K:\\Kira Simonson\\Stressor Densities\\MRB"

#merging files into one data frame for MRB
# Load all csv files in folder into list
data.pathMRB <- paste(basepathMRB, "\\Stressor Data\\", sep="")
filelistMRB <- list.files(path = data.pathMRB, pattern="*.csv")

# read in each .csv file in folder and create a data frame with the same
name as the .csv file
for (i in 1:length(filelistMRB)){
  assign(file_path_sans_ext(filelistMRB[i]),
         read.csv(paste(data.pathMRB, filelistMRB[i], sep='')) )}

# Get all MRB information relevant into one df using MRB FP HUC-12 as
basis
data.mergeMRB <- merge(FP_info_MRB[,c("HUC12", "HUC12_Areakm2",
"mx_StrO","pathtype")], NLCD_MRB[, c("HUC12",
"Developed1941","Developed1945","Developed1950","Developed1955","Developed
1960","Developed1965","Developed1970","Developed1975","Developed1980","Dev
eloped1985","Developed1990","Developed1995","Developed2000")], by =
"HUC12", all.x = TRUE)

data.mergeMRB <- merge(data.mergeMRB, NLCD_MRB[, c("HUC12",
"Ag1941","Ag1945","Ag1950","Ag1955","Ag1960","Ag1965","Ag1970","Ag1975","A
g1980","Ag1985","Ag1990","Ag1995","Ag2000")], by = "HUC12", all.x = TRUE)

#Buildings
data.mergeMRB <- merge(data.mergeMRB, Buildings[, c("HUC12", "area")], by
= "HUC12", all.x = TRUE)
colnames(data.mergeMRB)[colnames(data.mergeMRB)=="area"] <- "Buildings"


#Canal Ditch
data.mergeMRB <- merge(data.mergeMRB, CanalDitch[, c("HUC12",
"totalLength")], by = "HUC12", all.x = TRUE)
colnames(data.mergeMRB)[colnames(data.mergeMRB)=="totalLength"] <-
"Ditches"


#Forest Loss
data.mergeMRB <- merge(data.mergeMRB, ForestLoss[, c("HUC12", "Loss")], by
= "HUC12", all.x = TRUE)
colnames(data.mergeMRB)[colnames(data.mergeMRB)=="Loss"] <- "ForestLoss"
```

```
#HA
data.mergeMRB <- merge(data.mergeMRB, HA[, c("HUC12", "MH20avg")], by =
"HUC12", all.x = TRUE)
colnames(data.mergeMRB)[colnames(data.mergeMRB)=="MH20avg"] <- "HA"


#Percent Impervious
data.mergeMRB <- merge(data.mergeMRB, Impervious[, c("HUC12", "Density")],
by = "HUC12", all.x = TRUE)
colnames(data.mergeMRB)[colnames(data.mergeMRB)=="Density"] <-
"Impervious"


#Levee
data.mergeMRB <- merge(data.mergeMRB, Levees[, c("HUC12", "Density")], by
= "HUC12", all.x = TRUE)
colnames(data.mergeMRB)[colnames(data.mergeMRB)=="Density"] <-
"LeveedArea"

#Vegetation
data.mergeMRB <- merge(data.mergeMRB, Vegetation[, c("HUC12", "Density")],
by = "HUC12", all.x = TRUE)
colnames(data.mergeMRB)[colnames(data.mergeMRB)=="Density"] <-
"NonNativeVeg"


#Roads Railroads
data.mergeMRB <- merge(data.mergeMRB, RoadsRailroads[, c("HUC12",
"totalLength")], by = "HUC12", all.x = TRUE)
colnames(data.mergeMRB)[colnames(data.mergeMRB)=="totalLength"] <-
"Roads_Rail"


#Wells
data.mergeMRB <- merge(data.mergeMRB, Wells[, c("HUC12", "Count")], by =
"HUC12", all.x = TRUE)
colnames(data.mergeMRB)[colnames(data.mergeMRB)=="Count"] <- "Wells"



data.mergeMRB <- data.mergeMRB %>% mutate_at(vars(-("pathtype")),
function(x) as.numeric(as.character(x)))


# Version with zeros changed to NA
data.mergeMRB.NA <- data.mergeMRB
data.mergeMRB.NA[data.mergeMRB.NA==0] <- NA
# version with NA changed to zero
data.mergeMRB[is.na(data.mergeMRB)] <- 0

# Save as .csv file
out.pathMRB <- paste(basepathMRB, "\\MergedOutputs\\", sep="")
out.fileMRB <- paste(out.pathMRB, "Combined_Data_MRB.csv", sep="")
```

```
write.csv(data.mergeMRB, file = out.fileMRB, row.names = FALSE)


##-------------------------------------------------------------------------
---


# Correlation analysis (w zero instead of NA)
Correl <- cor(data.merge[,5:length(data.merge)], use =
"pairwise.complete.obs")
# Significance test
res <- cor.mtest(data.merge[,5:length(data.merge)], conf.level =0.95)
# Plotting (and saving) correlations


corplot<-ggcorrplot(Correl, hc.order = TRUE, type =
"upper",lab=TRUE,show.legend=FALSE,
           outline.col = "white",
           colors = c("#A63446", "white","#0C6291"))

ggsave("Correlation.pdf",plot=corplot,path=out.path,width = 12, height =
7.5, units = "in", dpi = 1000)

#MRB Correlation


# Correlation analysis (w zero instead of NA)
CorrelMRB <- cor(data.mergeMRB[,5:length(data.mergeMRB)], use =
"pairwise.complete.obs")

# Significance test
resMRB <- cor.mtest(data.mergeMRB[,5:length(data.merge)], conf.level
=0.95)


corplotMRB<-ggcorrplot(CorrelMRB, hc.order = TRUE, type =
"upper",lab=TRUE,show.legend=FALSE,
                    outline.col = "white",
                    colors = c("#0C6291", "white", "#A63446"))

ggsave("Correlation MRB.pdf",plot=corplotMRB,path=out.pathMRB,width = 12,
height = 7.5, units = "in", dpi = 1000)


#Calculation of IFI for US
#kira simonson
#spring 2022

# Take assembled stressor data and translate to 0 to 1 metrics
pacman::p_load(dplyr,ggplot2,tools,data.table,tidyr,emmeans,ggcorrplot,RCo
lorBrewer,psych)

# set basepath
basepath <- "K:\\Kira Simonson\\Stressor Densities"
```

```
out.path <- paste(basepath, "\\MergedOutputs\\", sep="") # for saving
outputs

# Load csv file of stressor data from "merge_stressor.R" script output
data.path <- paste(basepath, "/MergedOutputs/Combined_Data.csv", sep="")
all.data <- read.csv(data.path)
data.names <- colnames(all.data)

# Convert HUC-12 from numeric to character
all.data$HUC12 <- as.character(all.data$HUC12)

# # boxplots to look at range of data for each stressor
#
# for (i in 5:ncol(all.data)){
#   hist(all.data[,i], main = data.names[i])
#   boxplot(all.data[,i], main = data.names[i])
#   text(y=fivenum(all.data[,i]), labels =
round(fivenum(all.data[,i]),digits = 2), x = 0.75)}

#convert building area to a percentage
all.data$Buildings <-(all.data$Buildings)/(all.data$HUC12_Areakm2)/100

# Get only final stressors and HUC-12 Identifier into df
keep.columns <- c("Agriculture", "Buildings", "Ditches",
"Developed","ForestLoss","Impervious", "LeveedArea", "NonNativeVeg",
"Roads_Rail","Wells", "HA")
stressors <- all.data[, keep.columns]

# adjust stressors that are not 0 to 1
# convert percent to decimal
stressors$Impervious <- stressors$Impervious/100
stressors$Agriculture <- stressors$Agriculture/100
stressors$NonNativeVeg <- stressors$NonNativeVeg/100
stressors$Developed <-stressors$Developed/100
stressors$ForestLoss <-stressors$ForestLoss/100


#Scale count and line denisty by max value observed
stressors$Ditches <- stressors$Ditches/max(stressors$Ditches)
stressors$Roads_Rail <- stressors$Roads_Rail/max(stressors$Roads_Rail)
stressors$Wells <- stressors$Wells/max(stressors$Wells)

# Scale buildings to max building density
stressors$Buildings <- stressors$Buildings/max(stressors$Buildings)

# Compare all measures
#boxplot(stressors, use.cols = TRUE, ylab = 'Stressor Density')


# stressor_plot <- ggplot(melt(stressors),aes(variable,
value,fill=variable)) +
#   geom_boxplot(alpha=0.5)+
#    theme_linedraw() +
```

```
#   theme(legend.position="none")+
#   labs(x="",y="Density",col='') +
#   scale_x_discrete(name = "Stressor")+
#   scale_fill_manual(values=as.vector(ocean.phase(13)))+
#   theme(text = element_text(size=16), panel.grid.major.x =
#           element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank())
#


# Make negative
stressors.neg <- 1-stressors
#boxplot(stressors.neg, use.cols = T)

######


# choose datasets by function
# Flood reduction
FR.stressors <- c("ForestLoss", "Developed", "LeveedArea", "Roads_Rail")
# Buildings not included b/c of high correlation

# Groundwater regulation
GW.stressors <- c("Impervious", "Ditches", "Agriculture", "ForestLoss",
"Wells")

# Sediment Flux
SF.stressors <- c("HA", "Agriculture", "Roads_Rail", "ForestLoss")

# Organics and Solute regulation
OS.stressors <- c("HA", "ForestLoss", "Impervious", "Roads_Rail")


# Habitat provisioning
HP.stressors <- c("Roads_Rail", "HA", "NonNativeVeg", "Developed",
"Agriculture", "ForestLoss")

data.byfunction <- list(FR.stressors, GW.stressors, SF.stressors,
OS.stressors, HP.stressors)


# Cap stressors at 90th percentile
stressors.scaled <- stressors
percent.capped <- list()



# loop over stressors
for (i in 1:ncol(stressors)) {
  #find 90th percentile
  limit <- quantile(stressors[,i], probs = 0.90)


  # for non-zero 90th percentiles, compute as relative to 90th percentile
```

```
  if (limit != 0) {
    stressors.scaled[,i] <- stressors.scaled[,i]/limit

    # Count percentage of data being capped to one
    percent.capped[[i]] <-
sum(stressors.scaled[,i]>1)/nrow(stressors.scaled)
    # set values over 90th percentile to 1
    stressors.scaled[,i][stressors.scaled[,i]>1] <- 1
  }
  else {
    # if 90th percentile is 0, scale relative to max value
    stressors.scaled[,i] <-
      stressors.scaled[,i]/max(stressors.scaled[,i]) }
  percent.capped[[i]] <- 0
}

#checking which stressors have 90th percentile = 0 , change i 1-11
#limit <- quantile(stressors[,i], probs = 0.90)
#limit != 0

#***(Ditches, Levees, wells)

## Boxplot scaled stressors
#boxplot(stressors.scaled, use.cols = TRUE, ylab = 'Scaled Stressor
Density')


scaled_plot <- ggplot(melt(stressors.scaled),aes(variable, value)) +
  geom_boxplot(alpha=0.5)+
   theme_linedraw() +
  theme(legend.position="none")+
  labs(x="",y="Scaled Density",col='') +
  scale_x_discrete(name = "Stressor")+
  #scale_fill_manual(values=as.vector(ocean.delta(11)))+
  theme(text = element_text(size=13), panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank())


ggsave("Scaled Stressor boxplot.pdf", plot = scaled_plot, path ="K:\\Kira
Simonson\\Stressor Densities\\Results\\" , width = 12, height = 7.5, units
= "in", dpi = 72)




# Compute Index as average of stressors scaled for each function

# this creates a dataframe for with each of the 5 floodplain functions as
headers
Function.Index.Scaled <- data.frame(matrix(NA, nrow =
nrow(stressors.scaled), ncol = length(data.byfunction)))
```

```
#this takes 1 minus the mean value of the relevant stressor densities for
each function, calculating function index values
for (i in 1:length(data.byfunction)) {
  Function.Index.Scaled[,i] <- 1 -
rowMeans(stressors.scaled[,data.byfunction[[i]]])}

colnames(Function.Index.Scaled) <- c("Floods",
"Groundwater","Sediment","Organics/Solutes","Habitat")

# plot boxplot of index by function
function.plot.scaled <-
  ggplot(melt(Function.Index.Scaled),aes(variable, value,fill=variable)) +
  geom_boxplot(alpha=0.5)+
  theme_linedraw() +
  theme(legend.position="none")+
  labs(x="Floodplain Function",y="Integrity Index",col='') +
  scale_y_continuous(limits = c(0,1)) +
  scale_fill_manual(values=as.vector(ocean.delta(5)))+
  theme(text = element_text(size=13), panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank())+
  ggtitle("Index of Floodplain Integrity by Function, Scaled Stressors")

#ggsave("Scaled function IFI boxplot.pdf", plot = function.plot.scaled,
path ="K:\\Kira Simonson\\Stressor Densities\\Results\\" , width = 8,
height = 7.5, units = "in", dpi = 72)



# plot correlation of Indices

function.scaled.cor <- cor(Function.Index.Scaled, use =
"pairwise.complete.obs")

corplot<-ggcorrplot(function.scaled.cor, hc.order = TRUE, type =
"lower",lab=TRUE,lab_col="white",outline.col = "white", colors = c(
"#A63446", "white","#0C6291"),show.legend = FALSE)


ggsave("function correlation.pdf", plot = corplot, path ="K:\\Kira
Simonson\\Stressor Densities\\Results\\" , width = 8, height = 7.5, units
= "in", dpi = 72)


# Compute overall Index of floodplain Integrity
# Compute overall Index of floodplain Integrity, IFI.geomean is the name
of the column, apply is computing the geometric mean across each row aka
FP unit
IFI.scaled <- data.frame(IFI.geomean = apply(Function.Index.Scaled, 1,
function(x) geometric.mean(x)))
IFI.product.scaled <- data.frame(IFI.prod = apply(Function.Index.Scaled,
1, prod))
IFI.combine.scaled <- data.frame(IFI.scaled,IFI.product.scaled)
```

```
IFI.scaled.plot <- ggplot(melt(IFI.scaled),aes(variable,
value,fill=variable)) +
  geom_boxplot(aes(alpha=0.5)) +
  theme_linedraw() +
  theme(legend.position="none")+
  labs(x="",y="Index of Floodplain Integrity",col='') +
  scale_y_continuous(limits = c(0,1)) +
  scale_x_discrete(name = "", labels = "")+
  scale_fill_manual(values=as.vector(ocean.delta(3)))+
  theme(text = element_text(size=13), panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank())

IFI.scaled.plot


# Export to csv
HUC12 <- all.data$HUC12

combined.data <- data.frame(HUC12,Function.Index.Scaled,IFI.scaled)
IFI.outfile <- paste(out.path, "IFI_Scaled.csv", sep="")
write.csv(combined.data, file = IFI.outfile)

######################

#regional data
region.stressors <- data.frame(HUC12,stressors.scaled)

region.stressors$HUC12 <- as.character(region.stressors$HUC12)
region.stressors[1:32071,"HUC12"] <-
paste("0",region.stressors[1:32071,"HUC12"],sep="")

region.stressors$HUC4 <- substring(region.stressors$HUC12,1,4)

LA <- region.stressors[region.stressors$HUC4=="0803",]
west<-  region.stressors[region.stressors$HUC4=="1014",]
CA<- region.stressors[region.stressors$HUC4=="1804",]

LA[c(1,13)] = NULL
west[c(1,13)] = NULL
CA[c(1,13)] = NULL

LA_scaled_plot <- ggplot(melt(LA),aes(variable, value)) +
  geom_boxplot(alpha=0.5)+
  theme_linedraw() +
  theme(legend.position="none")+
  labs(x="",y="Scaled Density",col='') +
  scale_x_discrete(name = "Stressor")+
  #scale_fill_manual(values=as.vector(ocean.delta(11)))+
  theme(text = element_text(size=10), panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank(),axis.text.x = element_text(angle =
90, hjust = 1))+
  labs(tag = "c)")
```

```
west_scaled_plot <- ggplot(melt(west),aes(variable, value)) +
  geom_boxplot(alpha=0.5)+
  theme_linedraw() +
  theme(legend.position="none")+
  labs(x="",y="Scaled Density",col='') +
  scale_x_discrete(name = "Stressor")+
  #scale_fill_manual(values=as.vector(ocean.delta(11)))+
  theme(text = element_text(size=10), panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank(),axis.text.x = element_text(angle =
90, hjust = 1))+
  labs(tag = "b)")

CA_scaled_plot <- ggplot(melt(CA),aes(variable, value)) +
  geom_boxplot(alpha=0.5)+
  theme_linedraw() +
  theme(legend.position="none")+
  labs(x="",y="Scaled Density",col='') +
  scale_x_discrete(name = "Stressor")+
  #scale_fill_manual(values=as.vector(ocean.delta(11)))+
  theme(text = element_text(size=10), panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank(),axis.text.x = element_text(angle =
90, hjust = 1))+
  labs(tag = "a)")



All.regions <- grid.arrange(CA_scaled_plot,
west_scaled_plot,LA_scaled_plot, ncol = 2)
ggsave("scaled stressors by region.pdf", plot = All.regions, path =
out.path, width =7.5, height = 8, units = "in", dpi = 72)



#' Compile years for MRB NLCD data
#' @description : grabs a series of CSV and combines them into a single
file.
#' @param filePath : folder location of csv files to compile
#' @param fileName : name of the output file containing complied
information
#'
# install.packages("pacman")
pacman::p_load(sf,terra,dplyr, readr,tools,data.table)


## this is for compiling the NLCD data for each year into a single data
frame which will then be rescaled and merged with the other datasets to
calculate IFI values
compileOutput <- function(filePath, fileName,fileoutput){
  files <- list.files(filePath, pattern = ".csv", full.names = TRUE)
  for(i in seq_along(files)){
    f1 <- read_csv(files[i],progress = FALSE) %>%
      dplyr::mutate("FIRST_huc1" = as.character(FIRST_huc1))
```

```
    if(i == 1){
      df <- f1
    }else{
      df <- bind_rows(df, f1)
    }
  }
  ## remove for dupicated features
  lc2 <- df[!duplicated(df$FIRST_huc1), ]
  # write out feature.
  write_csv(x = lc2, file = paste0(fileoutput,"/",fileName,".csv"))
  # write_csv(x = df, file = paste0(fileoutput,"/",fileName,".csv"))
  }


#compile all outputs for each year
#1941 - 2000
MRB41 <- compileOutput(filePath= paste0("K:\\Kira Simonson\\Stressor
Densities\\MRB\\NLCD outputs 04 13 2022\\1941\\"),fileName=
"MRB1941",fileoutput="K:\\Kira Simonson\\Stressor Densities\\MRB\\NLCD
outputs 04 13 2022\\")

#1945 - 2000
for (i in 0:11){
  n = 1945+5*i
  compileOutput(filePath= paste0("K:\\Kira Simonson\\Stressor
Densities\\MRB\\NLCD outputs 04 13 2022\\",n, "\\"),fileName=
paste0("MRB",n),fileoutput="K:\\Kira Simonson\\Stressor
Densities\\MRB\\NLCD outputs 04 13 2022\\")
}

## merge all years into a single data frame

basepathMRB = "K:\\Kira Simonson\\Stressor Densities\\MRB"

# Load all csv files in folder into list
data.path <- paste(basepathMRB, "\\NLCD outputs 04 13 2022\\", sep="")
filelist <- list.files(path = data.path, pattern="*.csv")

# read in each .csv file in folder and create a data frame with the same
name as the .csv file
for (i in 1:length(filelist)){
  assign(file_path_sans_ext(filelist[i]),
         read.csv(paste(data.path, filelist[i], sep='')) )}

## Get all information relevant into one df using FP HUC-12 as basis
data.merge <- merge(FP_info_MRB[,c("HUC12", "HUC12_Areakm2",
"mx_StrO","pathtype")], MRB1941[, c("FIRST_huc1", "X2","X6")], by.x =
"HUC12", by.y = "FIRST_huc1", all.x = TRUE)
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1941","Ag1941"))

data.merge <- merge(data.merge, MRB1945[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)
```

```
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1945","Ag1945"))

data.merge <- merge(data.merge, MRB1950[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1950","Ag1950"))

data.merge <- merge(data.merge, MRB1955[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1955","Ag1955"))

data.merge <- merge(data.merge, MRB1960[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1960","Ag1960"))

data.merge <- merge(data.merge, MRB1965[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1965","Ag1965"))

data.merge <- merge(data.merge, MRB1970[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1970","Ag1970"))

data.merge <- merge(data.merge, MRB1975[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)

setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1975","Ag1975"))

data.merge <- merge(data.merge, MRB1980[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1980","Ag1980"))

data.merge <- merge(data.merge, MRB1985[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1985","Ag1985"))

data.merge <- merge(data.merge, MRB1990[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1990","Ag1990"))

data.merge <- merge(data.merge, MRB1995[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed1995","Ag1995"))
```

```
data.merge <- merge(data.merge, MRB2000[, c("FIRST_huc1", "X2","X6")],
by.x = "HUC12", by.y = "FIRST_huc1", all.x = TRUE)
setnames(data.merge, old =c("X2", "X6"), new =
c("Developed2000","Ag2000"))


# Save as .csv file
out.path <- paste(basepathMRB, "\\Stressor Data\\", sep="")
out.file <- paste(out.path, "NLCD_MRB.csv", sep="")
write.csv(data.merge, file = out.file, row.names = FALSE)

#Calculation of IFI for MRB
#kira simonson
#spring 2022

# Take assembled stressor data and translate to 0 to 1 metrics
pacman::p_load(dplyr,ggplot2,tools,data.table,tidyr,emmeans,corrplot,RColo
rBrewer,psych)

# set basepath
basepath <- "K:\\Kira Simonson\\Stressor Densities\\MRB"

out.path <- paste(basepath, "\\MergedOutputs\\", sep="") # for saving
outputs

# Load csv file of stressor data from "merge_stressor.R" script output
data.path <- paste(basepath, "/MergedOutputs/Combined_Data_MRB.csv",
sep="")
all.data <- read.csv(data.path)
data.names <- colnames(all.data)
# Convert HUC-12 from numeric to character
all.data$HUC12 <- as.character(all.data$HUC12)

# boxplots to look at range of data for each stressor

for (i in 5:ncol(all.data)){
  hist(all.data[,i], main = data.names[i])
  boxplot(all.data[,i], main = data.names[i])
  text(y=fivenum(all.data[,i]), labels =
round(fivenum(all.data[,i]),digits = 2), x = 0.75)}

#convert building area to a percentage
all.data$Buildings <-(all.data$Buildings)/(all.data$HUC12_Areakm2)/100


years <-
list("1941","1945","1950","1955","1960","1965","1970","1975","1980","1985"
,"1990","1995","2000")
for(n in 1:length(years)){

# Get only final stressors and HUC-12 Identifier into df
keep.columns <- c(paste("Ag",years[n],sep=""), "Buildings", "Ditches",
paste("Developed",years[n],sep=""),"ForestLoss","Impervious",
"LeveedArea", "NonNativeVeg", "Roads_Rail","Wells", "HA")
```

```
stressors <- all.data[, keep.columns]



# adjust stressors that are not 0 to 1
# convert percent to decimal
stressors$Impervious <- stressors$Impervious/100
stressors$NonNativeVeg <- stressors$NonNativeVeg/100
stressors$ForestLoss <-stressors$ForestLoss/100

#rescaling ag and developed for MRB
stressors[,1] <-stressors[,1]/100
stressors[,4] <-stressors[,4]/100


#Scale count and line denisty by max value observed
stressors$Ditches <- stressors$Ditches/max(stressors$Ditches)
stressors$Roads_Rail <- stressors$Roads_Rail/max(stressors$Roads_Rail)
stressors$Wells <- stressors$Wells/max(stressors$Wells)

# Scale buildings to max building density
stressors$Buildings <- stressors$Buildings/max(stressors$Buildings)

# Compare all measures
boxplot(stressors, use.cols = TRUE, ylab = 'Stressor Density')

# Make negative
stressors.neg <- 1-stressors
#boxplot(stressors.neg, use.cols = T)

######

# choose datasets by function
# Flood reduction
FR.stressors <- c("ForestLoss", paste("Developed",years[n],sep=""),
"LeveedArea","Roads_Rail")
#Buildings not included b/c of high correlation

# Groundwater regulation
GW.stressors <- c("Impervious", "Ditches", paste("Ag",years[n],sep=""),
"ForestLoss", "Wells")

# Sediment
SF.stressors <- c("HA", paste("Ag",years[n],sep=""), "Roads_Rail",
"ForestLoss")

# Organics and Solute regulation
OS.stressors <- c("HA", "ForestLoss", "Impervious", "Roads_Rail")


# Habitat provisioning
HP.stressors <- c("Roads_Rail", "HA", "NonNativeVeg",
paste("Developed",years[n],sep=""), paste("Ag",years[n],sep=""),
"ForestLoss")
```

```
data.byfunction <- list(FR.stressors, GW.stressors, SF.stressors,
OS.stressors, HP.stressors)


# Cap stressors at 90th percentile
stressors.scaled <- stressors
percent.capped <- list()

#checking which stressors have 90th percentile = 0 , change i 1-11
#limit <- quantile(stressors[,i], probs = 0.90)
#limit != 0


# loop over stressors
for (i in 1:ncol(stressors)) {
  #find 90th percentile
  limit <- quantile(stressors[,i], probs = 0.90)

  # for non-zero 90th percentiles, compute as relative to 90th percentile
  if (limit != 0) {
    stressors.scaled[,i] <- stressors.scaled[,i]/limit

    # Count percentage of data being capped to one
    percent.capped[[i]] <-
sum(stressors.scaled[,i]>1)/nrow(stressors.scaled)
    # set values over 90th percentile to 1
    stressors.scaled[,i][stressors.scaled[,i]>1] <- 1
  }
  else {
    # if 90th percentile is 0, scale relative to max value
    stressors.scaled[,i] <-
      stressors.scaled[,i]/max(stressors.scaled[,i]) }
  percent.capped[[i]] <- 0
}


## Boxplot scaled stressors
boxplot(stressors.scaled, use.cols = TRUE, ylab = 'Scaled Stressor
Density')

# Compute Index as average of stressors scaled for each function

# this creates a dataframe for with each of the 5 floodplain functions as
headers
Function.Index.Scaled <- data.frame(matrix(NA, nrow =
nrow(stressors.scaled), ncol = length(data.byfunction)))

#this takes 1 minus the mean value of the relevant stressor densities for
each function, calculating function index values
for (i in 1:length(data.byfunction)) {
  Function.Index.Scaled[,i] <- 1 -
rowMeans(stressors.scaled[,data.byfunction[[i]]])}
```

```
colnames(Function.Index.Scaled) <- c("Floods",
"Groundwater","Sediment","Organics/Solutes","Habitat")

# plot boxplot of index by function
function.plot.scaled <- ggplot(stack(Function.Index.Scaled), aes(x = ind,
y = values)) +
  geom_boxplot() +
  scale_y_continuous(limits = c(0,1)) +
  xlab("Floodplain Function") +
  ylab("Integrity Index") +
  ggtitle(paste("Index of Floodplain Integrity by Function, Scaled
Stressors",years[n]))

function.plot.scaled


# plot correlation of Indices
function.scaled.cor <- cor(Function.Index.Scaled, use =
"pairwise.complete.obs")
out.graph <- paste(out.path,
"IFI_Scaled_Correlation_",years[n],".jpg",sep="")
jpeg(out.graph, width = 2000, height = 2000, units = "px")
corrplot(function.scaled.cor, type = "upper", method = "circle",
tl.col="black", tl.srt=45,tl.cex= 4.5, diag=FALSE, addCoef.col =
"#bbbcc1", number.cex = 4, cl.pos ="n")
dev.off()

# Compute overall Index of floodplain Integrity
# IFI.geomean is the name of the column, apply is computing the geometric
mean across each row aka FP unit
IFI.scaled <- data.frame(IFI.geomean = apply(Function.Index.Scaled, 1,
function(x) geometric.mean(x)))
IFI.product.scaled <- data.frame(IFI.prod = apply(Function.Index.Scaled,
1, prod))
IFI.combine.scaled <- data.frame(IFI.scaled)

IFI.scaled.plot <- ggplot(stack(IFI.combine.scaled), aes(x = ind, y =
values)) +
  scale_y_continuous(limits = c(0,1)) +
  geom_boxplot() +
  xlab("") +
  ylab(paste("Index of Floodplain Integrity, scaled stressors",years[n]))

IFI.scaled.plot

# Export to csv
HUC12 <- all.data$HUC12

combined.data <-
data.frame(HUC12,Function.Index.Scaled,IFI.combine.scaled)
IFI.outfile <- paste(out.path, "IFI_Scaled",years[n],".csv", sep="")
write.csv(combined.data, file = IFI.outfile)

######################}
```

IFI analysis

The following R scripts were used to geospatially analyze the IFI dataset:

```
## IFI analysis

pacman::p_load(ggplot2,RColorBrewer,wesanderson,tidyr,grid,gridExtra,emmea
ns,scales,dplyr,data.table,stringr,psych,ggpubr,tidyverse,reshape,pals,too
ls,cmocean,viridis,moments,fmsb,ggridges,hrbrthemes)

# set paths
basepath <- "K:\\Kira Simonson\\Stressor Densities"
out.path <- paste(basepath, "\\Results\\", sep="")
out.path.thesis <-"K:\\Kira Simonson\\Thesis\\Figures\\"

#loading IFI data

all.data <- read.csv("K:\\Kira Simonson\\Stressor
Densities\\MergedOutputs\\IFI_Scaled.csv")
colnames(all.data)[which(names(all.data) == "IFI.geomean")] <-
"IFI_geomean"
col.names <- colnames(all.data)

FP_info <- read.csv(paste(basepath, "\\Stressor
Data\\FP_info1.csv",sep=""))

all.data <-
merge(all.data,FP_info[,c("FIRST_area","FIRST_huc1","COMID","FTYPE","StrmO
rd")],by.x = "HUC12", by.y="FIRST_huc1",all.x = TRUE)


all.data$X= NULL

#urban data

fp_urban <- read.csv("K:\\Kira Simonson\\Stressor Densities\\
urban_vs_non_urban.csv")
fp_urban <-fp_urban[,c("FIRST_huc1","urban")]

all.data <-left_join(all.data,fp_urban,by = c("HUC12" ="FIRST_huc1"))


#ecoregion data
fp_eco <- read.csv("K:\\Kira Simonson\\Stressor
Densities\\fp_unit_ecoregion.csv")
fp_eco <- fp_eco[,c("FIRST_huc1","WSA9")]

all.data <-left_join(all.data,fp_eco,by = c("HUC12"="FIRST_huc1"))



#IWI data
#get IWI data (version 1, original, version 2, linear/equal weights, 2.1,
rescaled/reweighted?)
```

```
#IWI_1 <- read.csv("K:\\Kira Simonson\\National Data Downloads\\IWI Data
(original and updated)\\IWI revision\\IWIICIgen_v1.csv")
#IWI_2 <- read.csv("K:\\Kira Simonson\\National Data Downloads\\IWI Data
(original and updated)\\IWI revision\\IWIICIgen_v2.csv")
#IWI_2.1 <- read.csv("K:\\Kira Simonson\\National Data Downloads\\IWI Data
(original and updated)\\IWI revision\\IWIICIgen_v2.1.csv")

#export IWI data to be aggregated in ArcMap

#IWI_all <- merge(IWI_1[,c("COMID","IWI")],IWI_2[,c("COMID","IWI2")],by =
"COMID",all.x = TRUE)
#IWI_all <-left_join(IWI_all,IWI_2.1[,c("COMID","IWI2.1")],by = "COMID")
#write.csv(IWI_all,"K:\\Kira Simonson\\National Data Downloads\\IWI Data
(original and updated)\\IWI_all_versions.csv")

#get fp intersected data (from arcgis)
fp_IWI <-read.csv("K:\\Kira Simonson\\National Data Downloads\\IWI Data
(original and updated)\\IWI_fp_join.csv")

fp_IWI <-fp_IWI %>% group_by(FIRST_1) %>% summarize(IWI1 = mean(IWI),IWI2
= mean(IWI2),IWI2.1 = mean(IWI2_1))
all.data <-left_join(all.data,fp_IWI,by = c("HUC12"="FIRST_1"))

all.data <-all.data[!duplicated(all.data$HUC12),]




# join HUC2 and HUC4 codes to HUC 12 codes
all.data$HUC12 <- as.character(all.data$HUC12)
all.data[1:32597,"HUC12"] <- paste("0",all.data[1:32597,"HUC12"],sep="")

all.data$HUC2 <- substring(all.data$HUC12,1,2)
all.data$HUC4 <- substring(all.data$HUC12,1,4)

setnames(all.data,old=c("FIRST_area","FTYPE","WSA9"),new=c("FP_Areakm2","p
athtype","ecoregion"))



write.csv(all.data,paste(out.path,"IFI.stats.csv"))

######
#function for displaying median

fun_median <- function(x){
  y = median(x,na.rm=T)
  return(data.frame(y=y,label=round(y,4)))
}

# general statistics about overall IFI
```

```
#
IFI.values <- all.data[, c("IFI_geomean","Floods", "Groundwater",
"Sediment", "Organics.Solutes", "Habitat")]

#summary table
IFI.stats <-melt(IFI.values) %>% group_by(variable)%>%
  summarize(min = min(value),
            p10 = quantile(value, 0.1),
            median = median(value),
            mean = mean(value),
            p90 = quantile(value, 0.9),
            max = max(value),
            sd = sd(value))

write.csv(IFI.stats,paste(out.path,"IFI.summary.stats.csv"))



########
# Histograms of IFI results

# find columns to plot
functions <- c("Floods", "Groundwater", "Sediment", "Organics.Solutes",
"Habitat")
func.IFI <- all.data[, functions]
colnames(func.IFI) <- c("Floods", "Groundwater", "Sediment",
"Organics_Solutes", "Habitat")


p <- ggplot(gather(func.IFI), aes(value)) +
  geom_histogram(bins = 20,fill="darkslategray3",color="#e9ecef",
alpha=0.9) +
  facet_wrap(~key, scales = 'free_y') +
  xlab("Index of Floodplain Integrity") +
  ylab("Count") +
  theme(text = element_text(size=20))


# save histogram
ggsave("IFI function histogram US.pdf", plot = p, path = out.path, width =
12, height = 7.5, units = "in", dpi = 72)




b <- ggplot(all.data, aes(x=IFI_geomean)) +
  geom_histogram(bins = 20,fill="darkslategray3",color="#e9ecef",
alpha=0.9) +
  xlab("Overall Index of Floodplain Integrity" ) +
  ylab("Count") +
  theme(text = element_text(size=20))

ggsave("IFI histogram US.pdf", plot = b, path = out.path, width = 12,
height = 7.5, units = "in", dpi = 72)
```

```
# ####
# # IFI by HUC 2
# all.data$HUC2 <- as.factor(all.data$HUC2)
# # get counts for label
# count.data <- as.data.frame((table(all.data$HUC2)))
# names(count.data)[1] = 'HUC2'
# count.data$Freq <- paste(" N =", as.character(count.data$Freq), sep = "
")

#stats for HUC2 (disregard this, it isn't weighted by area)
#
# HUC2_comparisons <- list( c("01","02"), c("02","03"), c("03", "04"),
c("04","05"), c("05","06"), c("06",
"07"),c("07","08"),c("08","09"),c("09","10"),c("10","11"),c("11","12"),c("
12","13"),c("13","14"),c("14","15"),c("15","16"),c("16","17"),c("17","18")
)
#
# HUC2 <- ggplot(na.omit(all.data),aes(HUC2, IFI_geomean, group = HUC2)) +
#   geom_boxplot(na.rm = TRUE,aes(fill=HUC2,alpha=0.3)) +
#   scale_x_discrete(name = "HUC2") + scale_y_continuous(limits =
c(0,1.5)) +
#   ylab("Overall IFI") +
#   theme_linedraw() +
#   theme(text = element_text(size=16), panel.grid.major.x =
#           element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank()) +
#   geom_text(data = count.data, aes(HUC2, y = 1.05, label = Freq),size =
3,position = position_dodge(width = 1)) +
#   scale_fill_manual(values=as.vector(ocean.delta(18)))+
#   theme(legend.position="none")+
#   stat_summary(fun.data = fun_median, geom="text",size=3,vjust=1.4)
#
#
#
#
# # boxplot
# ggsave("HUC2 US.pdf", plot = HUC2, path = out.path, width = 12, height =
7.5, units = "in", dpi = 72)
#
#
# #summary table
# HUC2.stats <-all.data %>%
#   group_by(HUC2) %>%
#   summarize(min = min(IFI_geomean),
#             p10 = quantile(IFI_geomean, 0.1),
#             median = median(IFI_geomean),
#             mean = mean(IFI_geomean),
#             p90 = quantile(IFI_geomean, 0.9),
#             max = max(IFI_geomean),
#             sd = sd(IFI_geomean))
#
# write.csv(HUC2.stats,paste(out.path,"HUC2.csv"))
```

```
#####################################

# getting data by HUC4, area weighted means

#summarize IFI data in each HUC4 by area-weighted means
by.area<- data.frame(Area_km2 = all.data$FP_Areakm2)

by.area$Floods <- func.IFI$Floods
by.area$Groundwater <- func.IFI$Groundwater
by.area$Sediment <- func.IFI$Sediment
by.area$Organics_Solutes <- func.IFI$Organics_Solutes
by.area$Habitat <- func.IFI$Habitat
by.area$Overall_IFI <- all.data$IFI_geomean
by.area$HUC4 <- all.data$HUC4

HUC4.sum <-by.area %>%
  group_by(HUC4) %>%
  summarise(areahuc4 = sum(Area_km2))

by.area <- left_join(by.area,HUC4.sum, by = c("HUC4"))

# by.area$weight <- by.area$Area_km2/by.area$areahuc4
#
# by.area$weighted_Floods <- by.area$weight * by.area$Floods
# by.area$weighted_Groundwater <- by.area$weight * by.area$Groundwater
# by.area$weighted_Sediment <- by.area$weight * by.area$Sediment
# by.area$weighted_Organics_Solutes <- by.area$weight *
by.area$Organics_Solutes
# by.area$weighted_Habitat <- by.area$weight * by.area$Habitat
# by.area$weighted_Overall_IFI <- by.area$weight * by.area$Overall_IFI
#
#
# HUC4.stats_area_weighted <-by.area %>%
#   group_by(HUC4) %>%
#   summarize(IFI = sum(weighted_Overall_IFI),
#             Floods = sum(weighted_Floods),
#             Groundwater = sum(weighted_Groundwater),
#             Sediment = sum(weighted_Sediment),
#             Organics.Solutes = sum(weighted_Organics_Solutes),
#             Habitat = sum(weighted_Habitat),
#             area_huc4 = mean(areahuc4),
#             count_fp_unit = n())
#
# selected.HUC4.stats_weighted <-
HUC4.stats_area_weighted[c(90,112,200),]
#
# write.csv(selected.HUC4.stats_weighted,paste(out.path,"HUC4
weighted.csv"))

HUC4.stats <-by.area %>%
  group_by(HUC4) %>%
  summarize(IFI = mean(Overall_IFI),
            Floods = mean(Floods),
```

```
            Groundwater = mean(Groundwater),
            Sediment = mean(Sediment),
            Organics.Solutes = mean(Organics_Solutes),
            Habitat = mean(Habitat),
            area_huc4 = mean(areahuc4),
            count_fp_unit = n())

selected.HUC4.stats <-  HUC4.stats[c(90,112,200),]

write.csv(selected.HUC4.stats,paste(out.path,"HUC4 selected stats.csv"))


#select
  #HUC 0803 for leveed areas LA
  #HUC 1014 unaltered west
  #HUC 1804 ag and development in CA



HUC4.chart <-
HUC4.stats[c(90,112,200),c("HUC4","IFI","Floods","Groundwater","Sediment",
"Organics.Solutes","Habitat")]

write.csv(HUC4.chart,paste(out.path,"HUC4 selected summaries.csv"))

###density plots

data.LA <- all.data[all.data$HUC4=="0803",]
setnames(data.LA,c('Floods','Groundwater','Sediment',"Organics.Solutes",'H
abitat','IFI_geomean'),c("Flood Reduction", "Groundwater Storage",
"Sediment Regulation", "Organics/Solutes Regulation", "Habitat
Provision","Overall"))
data.west <- all.data[all.data$HUC4=="1014",]
setnames(data.west,c('Floods','Groundwater','Sediment',"Organics.Solutes",
'Habitat','IFI_geomean'),c("Flood Reduction", "Groundwater Storage",
"Sediment Regulation", "Organics/Solutes Regulation", "Habitat
Provision","Overall"))
data.CA <- all.data[all.data$HUC4=="1804",]
setnames(data.CA,c('Floods','Groundwater','Sediment',"Organics.Solutes",'H
abitat','IFI_geomean'),c("Flood Reduction", "Groundwater Storage",
"Sediment Regulation", "Organics/Solutes Regulation", "Habitat
Provision","Overall"))


melt.LA <-melt(data.LA, measure.vars=c("Flood Reduction", "Groundwater
Storage", "Sediment Regulation", "Organics/Solutes Regulation", "Habitat
Provision","Overall"))
melt.west <-melt(data.west, measure.vars=c("Flood Reduction", "Groundwater
Storage", "Sediment Regulation", "Organics/Solutes Regulation", "Habitat
Provision","Overall"))
melt.CA <-melt(data.CA, measure.vars=c("Flood Reduction", "Groundwater
Storage", "Sediment Regulation", "Organics/Solutes Regulation", "Habitat
Provision","Overall"))
```

```
LA.plot <- ggplot(melt.LA, aes(x = value, y = variable,fill=..x..)) +
  geom_density_ridges_gradient() +
  theme_ridges() +
  scale_fill_gradientn(colours = c('#671733','#913a3e',  '#C16F4A',
      '#DDB96C'  ,'#F5EDA2', '#C3E09B',  '#91B892'
      ,'#5575A8','#493a87'),breaks=c(0,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1))+
  #labs(title = 'IFI values for Lower Mississippi-Yazoo') +
  ylab("")+
  scale_x_continuous(name="IFI Value", limits=c(0, 1))+
  ggtitle("c) Lower Mississippi-Yazoo")+
  theme(axis.text=element_text(size=10),
        axis.title=element_text(size=10),legend.position =
"none",panel.spacing = unit(0.1, "lines"),
        strip.text.x = element_text(size =
6),axis.text.y=element_blank(),plot.title = element_text(size = (12)))


LA.plot




west.plot <- ggplot(melt.west, aes(x = value, y = variable,fill=..x..)) +
  geom_density_ridges_gradient() +
  theme_ridges() +
  scale_fill_gradientn(colours = c('#671733','#913a3e',  '#C16F4A',
      '#DDB96C'  ,'#F5EDA2', '#C3E09B',  '#91B892'
      ,'#5575A8','#493a87'),breaks=c(0,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1))+
  #labs(title = 'IFI values for Missouri-White') +
  ylab("")+
  scale_x_continuous(name="IFI Value", limits=c(0, 1))+
  ggtitle("b) Missouri-White")+
  theme(axis.text=element_text(size=10),
        axis.title=element_text(size=10),legend.position =
"none",panel.spacing = unit(0.1, "lines"),
        strip.text.x = element_text(size =
6),axis.text.y=element_blank(),plot.title = element_text(size = (12)))


west.plot


CA.plot <- ggplot(melt.CA, aes(x = value, y = variable,fill=..x..)) +
  geom_density_ridges_gradient() +
  theme_ridges() +
  scale_fill_gradientn(colours = c('#671733','#913a3e',  '#C16F4A',
      '#DDB96C'  ,'#F5EDA2', '#C3E09B',  '#91B892'
      ,'#5575A8','#493a87'),breaks=c(0,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1))+
  #labs(title = 'IFI values for San Joaquin') +
  ylab("")+
  scale_x_continuous(name="IFI Value", limits=c(0, 1))+
  ggtitle("a) San Joaquin")+
```

```
  theme(axis.text=element_text(size=10),
        axis.title=element_text(size=10),legend.position =
"none",panel.spacing = unit(0.1, "lines"),
        strip.text.x = element_text(size =
6),axis.text.y=element_blank(),plot.title = element_text(size = (12)))


CA.plot


ggsave("Lower Mississippi-Yazoo plot.pdf", plot = LA.plot, path =
out.path, width = 6.5, height = 5, units = "in", dpi = 72)
ggsave("Missouri-White plot.pdf", plot = west.plot, path = out.path, width
= 6.5, height = 5, units = "in", dpi = 72)
ggsave("San Joaquin plot.pdf", plot = CA.plot, path = out.path, width =
6.5, height = 5, units = "in", dpi = 72)


All.regions.figure <- grid.arrange(CA.plot, west.plot,LA.plot, ncol = 3)
ggsave("all plots by region.pdf", plot = All.regions.figure, path =
out.path, width =10, height = 5, units = "in", dpi = 72)

ggsave("all plots by region.png", plot = All.regions.figure, path =
out.path.thesis, width =10, height = 5, units = "in", dpi = 1000)


LA.stats <- melt(data.LA[,c(18,2:7)]) %>% group_by(variable)%>%
  summarize(min = min(value),
            median = median(value),
            mean = mean(value),
            max = max(value),
            sd = sd(value))

west.stats <- melt(data.west[,c(18,2:7)]) %>% group_by(variable)%>%
  summarize(min = min(value),
            median = median(value),
            mean = mean(value),
            max = max(value),
            sd = sd(value))

CA.stats <- melt(data.CA[,c(18,2:7)]) %>% group_by(variable)%>%
  summarize(min = min(value),
            median = median(value),
            mean = mean(value),
            max = max(value),
            sd = sd(value))

write.csv(LA.stats,paste(out.path,"LA.stats.csv"))

write.csv(west.stats,paste(out.path,"west.stats.csv"))

write.csv(CA.stats,paste(out.path,"CA.stats.csv"))
```

```
# create radar plots
# HUC4.chart.LA <- rbind(rep(1,6) , rep(0,6) ,
HUC4.chart[1,c("IFI","Floods","Groundwater","Sediment","Organics.Solutes",
"Habitat")] )
# HUC4.chart.west <- rbind(rep(1,6) , rep(0,6) ,
HUC4.chart[2,c("IFI","Floods","Groundwater","Sediment","Organics.Solutes",
"Habitat")] )
# HUC4.chart.CA <- rbind(rep(1,6) , rep(0,6) ,
HUC4.chart[3,c("IFI","Floods","Groundwater","Sediment","Organics.Solutes",
"Habitat")] )
#
# radarchart(HUC4.chart.LA)
# radarchart(HUC4.chart.west)
# radarchart(HUC4.chart.CA)
#
# #polar coordinates
# # HUC4.data <- data.frame(variable=
c("IFI","Floods","Groundwater","Sediment","Organics.Solutes","Habitat") )
# # HUC4.data$LA <-
t(HUC4.stats[90,c("IFI","Floods","Groundwater","Sediment","Organics.Solute
s","Habitat")])
# # HUC4.data$CA <-
t(HUC4.stats[130,c("IFI","Floods","Groundwater","Sediment","Organics.Solut
es","Habitat")])
# # HUC4.data$CO <-
t(HUC4.stats[198,c("IFI","Floods","Groundwater","Sediment","Organics.Solut
es","Habitat")])
# #
# # plot.LA <- ggplot(HUC4.data,aes(x = variable,y = LA, fill =
factor(variable))) +
# #   geom_col(width = 1, color = "white") +
# #   coord_polar()
# #
# # plot.LA
```

```
###
# IFI by stream order
all.data[all.data == -999] <- NA
count.data <- as.data.frame((table(all.data$StrmOrd)))
names(count.data)[1] = 'StrmOrd'
count.data$Freq <- paste(" N =", as.character(count.data$Freq), sep = " ")

SO_comparisons <- list( c("1","2"), c("2","3"), c("3", "4"), c("4","5"),
c("5","6"), c("6", "7"),c("7","8"),c("8","9"),c("9","10"))

#reorder stream order IFI by median for a color gradient on boxplot
```

```
strm_value <-na.omit(all.data) %>% group_by(StrmOrd) %>% summarize(mean =
mean(IFI_geomean),median=median(IFI_geomean))
strm_value <- strm_value[order(strm_value$mean),]

strm_value$color <-ocean.oxy(10)
strm_value <- strm_value[order(strm_value$StrmOrd),]
strm_colors <- as.list(strm_value$color)
strm_value$mean<- substring(strm_value$mean,1,5)
strm_value$mean<- paste(" Mean =", as.character(strm_value$mean), sep = "
")
strm_value$median<- substring(strm_value$median,1,5)
strm_value$median<- paste(" Median =", as.character(strm_value$median),
sep = " ")


SO <- ggplot(na.omit(all.data), aes(StrmOrd, IFI_geomean, group =
StrmOrd)) +
  geom_boxplot(na.rm = TRUE,aes(fill=factor(StrmOrd),alpha=0.3)) +
  scale_x_discrete(name = "Stream Order", breaks = seq(1:10)) +
scale_y_continuous(limits = c(0,1.5)) +
  ylab("Overall IFI") +
  theme_linedraw() +
  theme_bw(base_size = 14) +
  theme(panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank()) +
  geom_text(data = count.data, aes(StrmOrd, y = 1.05, label = Freq),
nudge_y = 0.001, size = 2.75,check_overlap = TRUE) +
  #geom_text(data = strm_value, aes(StrmOrd, y = 1.05, label = mean),
nudge_y = 0.045, size = 3,check_overlap = TRUE) +
  #geom_text(data = strm_value, aes(StrmOrd, y = 1.05, label = median),
nudge_y = 0.001, size = 3,check_overlap = TRUE) +
   scale_fill_manual(values=c(strm_colors))+
  theme(legend.position="none")+
  stat_compare_means(comparisons = SO_comparisons, label = "p.signif",
method = "t.test",label.y = seq(1.10, 1.3,0.2/length(SO_comparisons)))+
  stat_summary(fun =mean, geom="point", shape=20, size=4, color="sienna4",
fill="papayawhip")
  #stat_summary(fun.data = mean, geom="text",size=3,vjust=1.45)

SO

# boxplot
ggsave("SO US.pdf", plot = SO, path = out.path, width = 10, height = 7.5,
units = "in", dpi = 72)

ggsave("SO US.png", plot = SO, path = out.path.thesis, width = 6.5, height
= 3.5, units = "in", dpi = 1000)



#summary table
SO.stats <-all.data %>%
  group_by(StrmOrd) %>%
```

```
  summarize(min = min(IFI_geomean),
            p10 = quantile(IFI_geomean, 0.1),
            median = median(IFI_geomean),
            mean = mean(IFI_geomean),
            p90 = quantile(IFI_geomean, 0.9),
            max = max(IFI_geomean),
            sd = sd(IFI_geomean))

write.csv(SO.stats,paste(out.path,"SO.stats.csv"))



###
# IFI by urban non urban
count.data <- as.data.frame((table(all.data$urban)))
names(count.data)[1] = 'urban'
count.data$Freq <- paste(" N =", as.character(count.data$Freq), sep = " ")

urban_comparisons <- list( c("Non-Urban","Urban"))


#pull mean and median  IFI by
urban_value <-na.omit(all.data) %>% group_by(urban) %>% summarize(mean =
mean(IFI_geomean),median=median(IFI_geomean),sd=sd(IFI_geomean))

urban_value$mean<- substring(urban_value$mean,1,5)
urban_value$mean<- paste(" Mean =", as.character(urban_value$mean), sep =
" ")
urban_value$median<- substring(urban_value$median,1,5)
urban_value$median<- paste(" Median =", as.character(urban_value$median),
sep = " ")

urban <- ggplot(na.omit(all.data), aes(urban, IFI_geomean, group = urban))
+
  geom_boxplot(na.rm = TRUE,aes(fill=factor(urban),alpha=0.3)) +
  scale_y_continuous(limits = c(0,1.5)) +
  scale_x_discrete(name = "", labels = c("Rural", "Urban"))+
  ylab("Overall IFI") +
  theme_linedraw() +
  theme_bw(base_size = 14) +
  theme(panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank()) +
  geom_text(data = count.data, aes(urban, y = 1.05, label = Freq), nudge_y
= 0.001, size = 3) +
  #geom_text(data = urban_value, aes(urban, y = 1.05, label = mean),
nudge_y = 0.045, size = 3.5,check_overlap = TRUE) +
  #geom_text(data = urban_value, aes(urban, y = 1.05, label = median),
nudge_y = 0.001, size = 3.5,check_overlap = TRUE) +
  scale_fill_manual(values=c("darkseagreen4","sienna3"))+
  theme(legend.position="none")+
  stat_compare_means(comparisons = urban_comparisons, label = "p.signif",
method = "t.test",label.y = c(1.15))+
```

```
  stat_summary(fun =mean, geom="point", shape=20, size=4, color="sienna4",
fill="papayawhip")
  #stat_summary(fun.data = fun_median, geom="text",size=3.5,vjust=1.4)

urban

# boxplot
ggsave("Urban_rural US.pdf", plot = urban, path = out.path, width = 5,
height =5, units = "in", dpi = 72)

ggsave("Urban_rural US.png", plot = urban, path = out.path.thesis, width =
5, height =4, units = "in", dpi = 1000)


#summary table
urban.stats <-all.data %>%
  group_by(urban) %>%
  summarize(min = min(IFI_geomean),
            p10 = quantile(IFI_geomean, 0.1),
            median = median(IFI_geomean),
            mean = mean(IFI_geomean),
            p90 = quantile(IFI_geomean, 0.9),
            max = max(IFI_geomean),
            sd = sd(IFI_geomean))

write.csv(urban.stats,paste(out.path,"urban_rural.stats.csv"))


# IFI by ecoregion
  #hex colors
    # Coastal Plains, Northern Appalachians, Northern Plains, Southern
Appalachians, Southern Plains, Temperate Plains, Upper Midwest, Western
Mountains, Xeric
    # c('#F2AC3D' , '#F1B6C4' . '#F2AB88' , '#C5F9C2' , '#CDCC75' ,
'#A9E447', '#8CDAF9','#A07225', '#BEC7F8')

count.data <- as.data.frame((table(all.data$ecoregion)))
names(count.data)[1] = 'ecoregion'
count.data$Freq <- paste(" N =", as.character(count.data$Freq), sep = " ")

#reorder stream order IFI by median for a color gradient on boxplot
eco_value <-na.omit(all.data) %>% group_by(ecoregion) %>% summarize(mean =
mean(IFI_geomean),median=median(IFI_geomean),sd=sd(IFI_geomean))
eco_value <- eco_value[order(eco_value$mean),]
eco_value$color <-ocean.oxy(9)
eco_colors <- as.list(eco_value$color)
eco_value$mean<- substring(eco_value$mean,1,5)
eco_value$mean<- paste(" Mean =", as.character(eco_value$mean), sep = " ")
eco_value$median<- substring(eco_value$median,1,5)
eco_value$median<- paste(" Median =", as.character(eco_value$median), sep
= " ")
```

```
ecoregion_comparisons <- list( c("NPL","XER"), c("XER","WMT"), c("WMT",
"UMW"), c("UMW","SPL"), c("SPL","NAP"), c("NAP",
"SAP"),c("SAP","TPL"),c("TPL","CPL"))


ecoregion <- ggplot(na.omit(all.data), aes(x=reorder(ecoregion,-
IFI_geomean), IFI_geomean, group = ecoregion)) +
  geom_boxplot(na.rm = TRUE,aes(fill=reorder(ecoregion,-
IFI_geomean),alpha=0.3)) +
  scale_y_continuous(limits = c(0,1.5)) +
  ylab("Overall IFI") +
  xlab("Ecoregion")+
  theme_linedraw() +
  theme_bw(base_size = 14) +
  theme(legend.position="none")+
  theme(panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank()) +
  #geom_text(data = count.data, aes(ecoregion, y = 1.05, label = Freq),
nudge_y = 0.001, size = 3) +
  geom_text(data = eco_value, aes(ecoregion, y = 1.05, label = mean),
nudge_y = 0.045, size = 3,check_overlap = TRUE) +
  geom_text(data = eco_value, aes(ecoregion, y = 1.05, label = median),
nudge_y = 0.001, size = 3,check_overlap = TRUE) +
  scale_fill_manual(values=ocean.oxy(9))+
  stat_compare_means(comparisons = ecoregion_comparisons, label =
"p.signif", method = "t.test",label.y = seq(1.10,
1.3,0.2/length(ecoregion_comparisons)))+
  stat_summary(fun =mean, geom="point", shape=20, size=4, color="sienna4",
fill="papayawhip")


ecoregion

# boxplot
ggsave("ecoregion US.pdf", plot = ecoregion, path = out.path, width = 10,
height = 7.5, units = "in", dpi = 72)


ggsave("ecoregion US.png", plot = ecoregion, path = out.path.thesis, width
= 6.5, height = 3.5, units = "in", dpi = 1000)

#summary table
ecoregion.stats <-all.data %>%
  group_by(ecoregion) %>%
  summarize(min = min(IFI_geomean),
            p10 = quantile(IFI_geomean, 0.1),
            median = median(IFI_geomean),
            mean = mean(IFI_geomean),
            p90 = quantile(IFI_geomean, 0.9),
            max = max(IFI_geomean),
            sd = sd(IFI_geomean))

write.csv(ecoregion.stats,paste(out.path,"ecoregion.stats.csv"))
```

```
#####
# IFI function to overall ratio
func.ratio <- apply(func.IFI, 2, function(x) x/all.data$IFI_geomean)
# Plot boxplots
ratio.df <- melt(func.ratio)
ratio.df <- ratio.df[!is.infinite(ratio.df$value),]
levels(ratio.df$X2) = c("Flood Reduction", "Groundwater Storage",
"Sediment Regulation","Organics/Solutes Regulation", "Habitat Provision")
ratio.plot <- ggplot(ratio.df, aes(x = X2, y = value)) +
  geom_boxplot() +
  theme_bw(base_size=14)+
  geom_hline(yintercept = 1, linetype = "dashed", size = 1) +
  scale_x_discrete(labels = function(x) str_wrap(x, width = 10)) +
  labs( y = "Ratio of Function to Overall IFI\n", x = NULL) +
  theme_linedraw() +
  theme(panel.grid.major.x = element_blank(), panel.grid.major.y =
element_blank(), panel.grid.minor.y = element_blank(),legend.position =
"none")
ratio.plot

#save
ggsave("function to overall ratio.pdf", plot = ratio.plot, path =
out.path, width = 7.5, height = 5, units = "in", dpi = 100)

ggsave("function to overall ratio.png", plot = ratio.plot, path =
out.path.thesis, width = 6.5, height = 3, units = "in", dpi = 1000)



# Test for significant differences

ratio.lm <- lm(value ~ X2, data = ratio.df)
ratio.pairwise <- lsmeans(ratio.lm, pairwise ~ X2)
method.contrasts <- ratio.pairwise$contrasts
method.contrasts
# results: all significantly different except sediment and organics

#######################################
# Plot bar graphs of function IFI by area

# Arrange data into groups of 0.05 bins
breaks <- seq(0.00, 1, 0.05)
breaks[1] <- -Inf

by.area<- data.frame(Area_km2 = all.data$FP_Areakm2)

by.area$Floods <- cut(func.IFI$Floods, breaks)
by.area$Groundwater <- cut(func.IFI$Groundwater, breaks)
by.area$Sediment <- cut(func.IFI$Sediment, breaks)
by.area$Organics_Solutes <- cut(func.IFI$Organics_Solutes, breaks)
by.area$Habitat <- cut(func.IFI$Habitat, breaks)
by.area$Overall_IFI <- cut(all.data$IFI_geomean, breaks)
```

```
Flood.sum <- by.area %>%
  group_by(Floods) %>%
  summarise(area = sum(Area_km2))

GW.sum <- by.area %>%
  group_by(Groundwater) %>%
  summarise(area = sum(Area_km2))

Sed.sum <- by.area %>%
  group_by(Sediment) %>%
  summarise(area = sum(Area_km2))

OS.sum <- by.area %>%
  group_by(Organics_Solutes) %>%
  summarise(area = sum(Area_km2))

Habitat.sum <- by.area %>%
  group_by(Habitat) %>%
  summarise(area = sum(Area_km2))

Overall.sum <- by.area %>%
  group_by(Overall_IFI) %>%
  summarise(area = sum(Area_km2))


# THIS IS ALL MANUAL
area.sum <- data.frame(breaks = OS.sum$Organics_Solutes)
area.sum$Floods <- Flood.sum$area
area.sum$Groundwater <- NA
area.sum[8:20,3] <- GW.sum$area
area.sum$Sediment <- Sed.sum$area
area.sum$Organics_Solutes <- OS.sum$area
area.sum$Habitat <- NA
area.sum[2:20,6] <- Habitat.sum$area
area.sum$Overall <- Overall.sum$area

area.sum$breaks <- as.numeric(area.sum$breaks)

# Graph with facet wrap "histograms"

area.df <- melt(area.sum, id = 1, measure = 2:7)
levels(area.df$variable) = c("Flood Reduction", "Groundwater Storage",
"Sediment Regulation",
                             "Organics/Solutes Regulation", "Habitat
Provision", "Overall IFI")

area.plot <- ggplot(data = na.omit(area.df), aes(x = breaks, y = value)) +
  geom_histogram(stat = "identity", bins = 20,  position =
position_nudge(x = -0.5), fill="darkseagreen4",color="#e9ecef", alpha=0.9)
+
  scale_x_continuous(limits = c(0, 20), breaks = seq(0, 20, 4),
                     labels = c("0", "0.2", "0.4", "0.6", "0.8", "1.0")) +
  facet_wrap(~ variable, ncol = 3) +
```

```
  labs(x = "IFI Value", y = bquote("Total floodplain area, " ~km^2)) +
  theme_bw(base_size = 12) +
  theme(strip.background =element_rect(fill="grey93"))
area.plot


ggsave("IFI histogram by area US.pdf", plot = area.plot, path = out.path,
width = 12, height = 7.5, units = "in", dpi = 72)

ggsave("IFI histogram by area US.png", plot = area.plot, path =
out.path.thesis, width = 6.5, height = 4.5, units = "in", dpi = 1000)


 ######################################

#IFI vs IWI comparison

# fit linear models
IWI_1.lm <- lm(data = all.data, IFI_geomean~ IWI1)
R2.IWI_1 <- summary(IWI_1.lm)$r.squared



cor.test(all.data$IWI1, all.data$IFI_geomean, method = c("pearson"))
summary(IWI_1.lm )

# Scatter plot of IWI_1 vs IFI
IWI_1.plot <- ggplot(all.data, aes(x = IWI1, y = IFI_geomean)) +
geom_point(size = 1)+
  xlim(0,1) + ylim(0,1) +
  coord_equal() +
  theme_bw(base_size=10) +
  xlab("IWI V1") +
  ylab("IFI") +
  geom_smooth(method=lm) +
  #geom_text(x= 0.1, y=0.05, label = paste0("R^2 =
",round(R2.IWI_1,2)),nudge_y =-0.25) +

  theme(text = element_text(size=10))+
  labs(tag = "a)")

IWI_1.plot

ggsave("IWI 1 plot.pdf", plot = IWI_1.plot, path = out.path, width = 12,
height = 7.5, units = "in", dpi = 72)


# fit linear models
IWI2.lm <- lm(data = all.data, IFI_geomean~ IWI2)
R2.IWI2 <- summary(IWI2.lm)$r.squared

cor.test(all.data$IWI2, all.data$IFI_geomean, method = c("pearson"))
summary(IWI2.lm )
```

```
# Scatter plot of IWI_1 vs IFI
IWI2.plot <- ggplot(all.data, aes(x = IWI2, y = IFI_geomean)) +
geom_point(size = 1)+
  xlim(0,1) + ylim(0,1) +
  coord_equal() +
  theme_bw(base_size=10) +
  xlab("IWI V2") +
  ylab(" ") +
  geom_smooth(method=lm) +
  #geom_text(x= 0.1, y=0.05, label = paste0("R^2 =
",round(R2.IWI2,2)),nudge_y =-0.25) +
  theme_bw() +
  theme(text = element_text(size=10))+
  labs(tag = "b)")

IWI2.plot

ggsave("IWI 2 plot.pdf", plot = IWI2.plot, path = out.path, width = 12,
height = 7.5, units = "in", dpi = 72)



# fit linear models
IWI2.1.lm <- lm(data = all.data, IFI_geomean~ IWI2.1)
R2.IWI2.1 <- summary(IWI2.1.lm)$r.squared

cor.test(all.data$IWI2.1, all.data$IFI_geomean, method = c("pearson"))
summary(IWI2.1.lm )

# Scatter plot of IWI_1 vs IFI
IWI2.1plot <- ggplot(all.data, aes(x = IWI2.1, y = IFI_geomean)) +
geom_point(size = 1)+
  xlim(0,1) + ylim(0,1) +
  coord_equal() +
  theme_bw(base_size=10) +
  xlab("IWI V2.1") +
  ylab(" ") +
  geom_smooth(method=lm) +
  #geom_text(x= 0.1, y=0.05, label = paste0("R^2 = ",round(R2.IWI2.1,2)))
+
  theme_bw() +
  theme(text = element_text(size=10))+
  labs(tag = "c)")

IWI2.1plot

ggsave("IWI 2.1 plot.pdf", plot = IWI2.1plot, path = out.path, width = 12,
height = 7.5, units = "in", dpi = 72)

All.IWI.figure <- grid.arrange(IWI_1.plot, IWI2.plot,IWI2.1plot, ncol = 3)

ggsave("All.IWI.figure.pdf", plot = All.IWI.figure, path = out.path, width
=12, height = 4, units = "in", dpi = 72)
```

```
ggsave("All.IWI.figure.png", plot = All.IWI.figure, path =
out.path.thesis, width =6.5, height = 2, units = "in", dpi = 1000)

##IFI MRB for individual years, done as a loop
#kira simonson
#spring 2022

## IFI analysis for MRB, this matches the overall IFI script, run for each
year


pacman::p_load(ggplot2,RColorBrewer,wesanderson,tidyr,grid,gridExtra,emmea
ns,scales,dplyr,data.table,stringr,psychTools,psych,ggpubr,tidyverse,tools
)

# set paths
basepath <- "K:\\Kira Simonson\\Stressor Densities\\MRB"
out.path <- paste(basepath, "\\Results\\", sep="")

#loading IFI data
# Load all csv files in folder into list
data.path <- paste(basepath, "\\MergedOutputs\\", sep="")


years <-
list("1941","1945","1950","1955","1960","1965","1970","1975","1980","1985"
,"1990","1995","2000")

# need to manually change n to save data

for(n in 1:length(years)){
  all.data <-
read.csv(paste(data.path,"IFI_Scaled",years[n],".csv",sep=""))

  colnames(all.data)[which(names(all.data) == "IFI.geomean")] <-
"IFI_geomean"
  col.names <- colnames(all.data)


  FP_info_MRB <- read.csv(paste(basepath, "\\Stressor
Data\\FP_info_MRB.csv",sep=""))

  all.data <-
merge(all.data,FP_info_MRB[,c("HUC12_Areakm2","HUC12","pathtype","mx_StrO"
)],by.x = "HUC12",all.x = TRUE)
  all.data$X= NULL

  # join HUC2 codes to HUC 12 codes
  all.data$HUC12 <- as.character(all.data$HUC12)
  all.data[1:14661,"HUC12"] <- paste("0",all.data[1:14661,"HUC12"],sep="")

  all.data$HUC2 <- substring(all.data$HUC12,1,2)

  #join HUC to huc4
```

```
  all.data$HUC4 <- substring(all.data$HUC12,1,4)

  #join HUC to huc6
  all.data$HUC6 <- substring(all.data$HUC12,1,6)


setnames(all.data,old=c("HUC12_Areakm2","mx_StrO"),new=c("FP_Areakm2","Str
mOrd"))


  ######
  # general statistics about overall IFI
  IFI.values <- all.data[, c("IFI_geomean","Floods", "Groundwater",
"Sediment", "Organics.Solutes", "Habitat")]
  IFI.stats <- describe(IFI.values,fast=TRUE)
  IFI.stats <- as.data.frame(IFI.stats)
  write.csv(IFI.stats,paste(out.path,years[n],"IFI.stats.csv"))


  ########
  # IFI results

  # find columns to plot
  functions <- c("Floods", "Groundwater", "Sediment", "Organics.Solutes",
"Habitat")
  func.IFI <- all.data[, functions]
  colnames(func.IFI) <- c("Floods", "Groundwater", "Sediment",
"Organics_Solutes", "Habitat")


  p <- ggplot(gather(func.IFI), aes(value)) +
    geom_histogram(bins = 20) +
    facet_wrap(~key, scales = 'free_y') +
    xlab("Index of Floodplain Integrity") +
    ylab("Count") +
    theme(text = element_text(size=20))

  p


  b <- ggplot(all.data, aes(x=IFI_geomean)) +
    geom_histogram(bins = 20) +
    xlab("Overall Index of Floodplain Integrity" ) +
    ylab("Count") +
    theme(text = element_text(size=20))
  b

  ####
  # IFI by HUC 2
  all.data$HUC2 <- as.factor(all.data$HUC2)
  # get counts for label
  count.data.HUC2 <- as.data.frame((table(all.data$HUC2)))
  names(count.data.HUC2)[1] = 'HUC2'
```

```
  count.data.HUC2$Freq <- paste(" N =",
as.character(count.data.HUC2$Freq), sep = " ")

  #stats for HUC2
  HUC2 <- ggplot(na.omit(all.data),aes(HUC2, IFI_geomean, group = HUC2)) +
    geom_boxplot(na.rm = TRUE) +
    scale_x_discrete(name = "HUC2") + scale_y_continuous(limits =
c(0,1.5)) +
    ylab(paste(years[n],"Overall IFI")) +
    theme_linedraw() +
    theme(text = element_text(size=16), panel.grid.major.x =
            element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank()) +
    geom_text(data = count.data.HUC2, aes(HUC2, y = 1.05, label = Freq),
nudge_y = 0.05, size = 3) +
    labs(tag = "c)")

  HUC2

  ggsave(paste(years[n],"HUC2_summary.png", sep=""), plot = HUC2, path =
out.path, width = 6.5, height = 7.5, units = "in", dpi = 1000)


  #summary table
  HUC2.stats <-all.data %>%
    group_by(HUC2) %>%
    summarize(min = min(IFI_geomean),
              p10 = quantile(IFI_geomean, 0.1),
              median = median(IFI_geomean),
              mean = mean(IFI_geomean),
              p90 = quantile(IFI_geomean, 0.9),
              max = max(IFI_geomean),
              sd = sd(IFI_geomean))

  write.csv(HUC2.stats,paste(out.path,years[n],"HUC2.csv"))

  ###
  # IFI by stream order
  all.data[all.data == -999] <- NA
  count.data <- as.data.frame((table(all.data$StrmOrd)))
  names(count.data)[1] = 'StrmOrd'
  count.data$Freq <- paste(" N =", as.character(count.data$Freq), sep = "
")

  SO_comparisons <- list( c("1","2"), c("2","3"), c("3", "4"), c("4","5"),
c("5","6"), c("6", "7"),c("7","8"),c("8","9"),c("9","10"))


  SO <- ggplot(na.omit(all.data), aes(StrmOrd, IFI_geomean, group =
StrmOrd)) +
    geom_boxplot(na.rm = TRUE) +
    scale_x_discrete(name = "Stream Order", breaks = seq(1:10)) +
scale_y_continuous(limits = c(0,1.5)) +
    ylab(paste(years[n],"Overall IFI")) +
```

```
    theme_linedraw() +
    theme(text = element_text(size=16), panel.grid.major.x =
            element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank()) +
    geom_text(data = count.data, aes(StrmOrd, y = 1.05, label = Freq),
nudge_y = 0.05, size = 4) +
    labs(tag = "c)") +
    stat_compare_means(comparisons = SO_comparisons, label = "p.signif",
method = "t.test",label.y = seq(1.25, 1.45,0.2/length(SO_comparisons)))

  SO



  #summary table
  SO.stats <-all.data %>%
    group_by(StrmOrd) %>%
    summarize(min = min(IFI_geomean),
              p10 = quantile(IFI_geomean, 0.1),
              median = median(IFI_geomean),
              mean = mean(IFI_geomean),
              p90 = quantile(IFI_geomean, 0.9),
              max = max(IFI_geomean),
              sd = sd(IFI_geomean))

  write.csv(SO.stats,paste(out.path,years[n],"_SO.stats.csv",sep=""))



  ####
  # IFI by HUC 4
  all.data$HUC4 <- as.factor(all.data$HUC4)
  # get counts for label
  count.data.HUC4 <- as.data.frame((table(all.data$HUC4)))
  names(count.data.HUC4)[1] = 'HUC4'
  count.data.HUC4$Freq <- paste(" N =",
as.character(count.data.HUC4$Freq), sep = " ")

  #stats for HUC4
  HUC4 <- ggplot(na.omit(all.data),aes(HUC4, IFI_geomean, group = HUC4)) +
    geom_boxplot(na.rm = TRUE) +
    scale_x_discrete(name = "HUC4") + scale_y_continuous(limits =
c(0,1.5)) +
    ylab(paste(years[n],"Overall IFI")) +
    theme_linedraw() +
    theme(text = element_text(size=16), panel.grid.major.x =
            element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank())
  HUC4

  #summary table
  HUC4.stats <-all.data %>%
    group_by(HUC4) %>%
    summarize(min = min(IFI_geomean),
```

```
              p10 = quantile(IFI_geomean, 0.1),
              median = median(IFI_geomean),
              mean = mean(IFI_geomean),
              p90 = quantile(IFI_geomean, 0.9),
              max = max(IFI_geomean),
              sd = sd(IFI_geomean))

  write.csv(HUC4.stats,paste(out.path,years[n],"HUC4.csv"))


  # IFI by HUC 6
  all.data$HUC6 <- as.factor(all.data$HUC6)
  # get counts for label
  count.data.HUC6 <- as.data.frame((table(all.data$HUC6)))
  names(count.data.HUC6)[1] = 'HUC6'
  count.data.HUC6$Freq <- paste(" N =",
as.character(count.data.HUC6$Freq), sep = " ")

  #stats for HUC6
  HUC6 <- ggplot(na.omit(all.data),aes(HUC6, IFI_geomean, group = HUC6)) +
    geom_boxplot(na.rm = TRUE) +
    scale_x_discrete(name = "HUC6") +
    scale_y_continuous(limits = c(0,1.5)) +
    ylab(paste(years[n],"Overall IFI")) +
    theme_linedraw() +
    theme(text = element_text(size=16), panel.grid.major.x =
            element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank(),axis.text.x = element_blank())

  HUC6



  #summary table
  HUC6.stats <-all.data %>%
    group_by(HUC6) %>%
    summarize(min = min(IFI_geomean),
              p10 = quantile(IFI_geomean, 0.1),
              median = median(IFI_geomean),
              mean = mean(IFI_geomean),
              p90 = quantile(IFI_geomean, 0.9),
              max = max(IFI_geomean),
              sd = sd(IFI_geomean))

  write.csv(HUC6.stats,paste(out.path,years[n],"HUC6.csv"))


}

## IFI analysis for MRB years, done just for overall IFI to compare trends
over period

pacman::p_load(ggplot2,RColorBrewer,reshape2,tidyr,grid,gridExtra,emmeans,
scales,dplyr,data.table,stringr,psych,ggpubr,tidyverse,tools)
```

```
# set paths
basepath <- "K:\\Kira Simonson\\Stressor Densities\\MRB"
out.path <- paste(basepath, "\\Results\\", sep="")
out.path.thesis <-"K:\\Kira Simonson\\Thesis\\Figures\\"

#loading IFI data
# Load all csv files in folder into list
data.path <- paste(basepath, "\\MergedOutputs\\", sep="")
filelist <- list.files(path = data.path, pattern="*.csv")

# read in each .csv file in folder and create a data frame with the same
name as the .csv file
for (i in 1:length(filelist)){
  assign(file_path_sans_ext(filelist[i]),
         read.csv(paste(data.path, filelist[i], sep='')) )}


FP_info_MRB <- read.csv(paste(basepath, "\\Stressor
Data\\FP_info_MRB.csv",sep=""))


all.data.MRB <-
merge(FP_info_MRB,IFI_Scaled1941[,c("HUC12","IFI.geomean")],by="HUC12")
all.data.MRB$HUC12_Name= NULL
setnames(all.data.MRB,old=c("HUC12_Areakm2","IFI.geomean"),new=c("FP_Areak
m2","IFI_1941"))

#1945
all.data.MRB <- merge(all.data.MRB, IFI_Scaled1945[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)
colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_1945"
#1950
all.data.MRB <- merge(all.data.MRB, IFI_Scaled1950[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)
colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_1950"
#1955
all.data.MRB <- merge(all.data.MRB, IFI_Scaled1955[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)
colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_1955"
#1960
all.data.MRB <- merge(all.data.MRB, IFI_Scaled1960[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)
colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_1960"
#1965
all.data.MRB <- merge(all.data.MRB, IFI_Scaled1965[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)
colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_1965"
#1970
```

```
all.data.MRB <- merge(all.data.MRB, IFI_Scaled1970[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)
colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_1970"
#1975
all.data.MRB <- merge(all.data.MRB, IFI_Scaled1975[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)
colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_1975"
#1980
all.data.MRB <- merge(all.data.MRB, IFI_Scaled1980[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)

colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_1980"
#1985
all.data.MRB <- merge(all.data.MRB, IFI_Scaled1985[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)
colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_1985"
#1990
all.data.MRB <- merge(all.data.MRB, IFI_Scaled1990[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)
colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_1990"
#1995
all.data.MRB <- merge(all.data.MRB, IFI_Scaled1995[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)
colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_1995"
#2000
all.data.MRB <- merge(all.data.MRB, IFI_Scaled2000[,c("HUC12",
"IFI.geomean")], by = "HUC12", all.x = TRUE)
colnames(all.data.MRB)[colnames(all.data.MRB)=="IFI.geomean"] <-
"IFI_2000"

# join HUC2 codes to HUC 12 codes
all.data.MRB$HUC12 <- as.character(all.data.MRB$HUC12)
all.data.MRB[1:14661,"HUC12"] <-
paste("0",all.data.MRB[1:14661,"HUC12"],sep="")

all.data.MRB$HUC2 <- substring(all.data.MRB$HUC12,1,2)
all.data.MRB$HUC2 <- as.character(all.data.MRB$HUC2)
#write.csv(all.data.MRB,paste(out.path,"all.IFI.MRB.csv"))

IFI.by.year <- all.data.MRB
IFI.by.year$HUC12 =NULL
IFI.by.year$FP_Areakm2=NULL
IFI.by.year$pathtype=NULL
IFI.by.year$mx_StrO=NULL
IFI.by.year$HUC2=NULL

#boxplot(IFI.by.year)
```

```
fun_median <- function(x){
  y = median(x,na.rm=T)
  return(data.frame(y=y,label=round(y,4)))
}


year_comparisons <-
list(c("IFI_1941","IFI_1945"),c("IFI_1945","IFI_1950"),c("IFI_1950","IFI_1
955"),c("IFI_1955","IFI_1960"),c("IFI_1960","IFI_1965"),c("IFI_1965","IFI_
1970"),c("IFI_1970","IFI_1975"),c("IFI_1975","IFI_1980"),c("IFI_1980","IFI
_1985"),c("IFI_1985","IFI_1990"),c("IFI_1990","IFI_1995"),c("IFI_1995","IF
I_2000"))

IFI_value <-melt(IFI.by.year) %>% group_by(variable) %>% summarize(mean =
mean(value),median=median(value))

IFI_value$mean<- substring(IFI_value$mean,1,5)
IFI_value$mean<- paste(" Mean =", as.character(IFI_value$mean), sep = " ")

IFI_value$median<- substring(IFI_value$median,1,5)
IFI_value$median<- paste(" Median =", as.character(IFI_value$median), sep
= " ")


p0 <- ggplot(melt(IFI.by.year),aes(variable, value,fill=variable)) +
  geom_boxplot(alpha=0.5)+
  theme_bw(base_size=12)+
  stat_summary(fun=mean, geom="point", shape=20, size=4, color="sienna4",
fill="papayawhip") +
  #geom_text(data = IFI_value, aes(variable, y = 1.05, label = mean),
nudge_y = 0.045, size = 3.5,check_overlap = TRUE) +
  #geom_text(data = IFI_value, aes(variable, y = 1.05, label = median),
nudge_y = 0.001, size = 3.5,check_overlap = TRUE) +
  theme_linedraw() +
  theme(legend.position="none")+
  labs(x="Year",y="Overall IFI",col='') +
  scale_x_discrete(labels =
c("1941","1945","1950","1955","1960","1965","1970","1975","1980","1985","1
990","1995","2000"))+
  scale_fill_manual(values=
c("darkseagreen4","darkseagreen4","darkseagreen4","darkseagreen4","darksea
green4","darkseagreen4","darkseagreen4","darkseagreen4","darkseagreen4","d
arkseagreen4","darkseagreen4","darkseagreen4","darkseagreen4","darkseagree
n4"))+
  theme( panel.grid.major.x =
           element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank()) +
  stat_compare_means(comparisons = year_comparisons, label = "p.signif",
method = "t.test",label.y = seq(1.05, 1.25, 0.2/length(year_comparisons)),
size = 3)

p0
```

```
# save plot
ggsave("IFI by year boxplot.pdf", plot = p0, path = out.path, width = 10,
height = 7.5, units = "in", dpi = 72)

ggsave("IFI by year boxplot.png", plot = p0, path = out.path.thesis, width
= 6.5, height = 3, units = "in", dpi = 1000)

#looking at change in function IFI over time


function.data.MRB <-
merge(FP_info_MRB,IFI_Scaled1941[,c(2,(3:7))],by="HUC12")
function.data.MRB$HUC12_Name= NULL
setnames(function.data.MRB,old=c("HUC12_Areakm2","Floods","Groundwater","S
ediment","Organics.Solutes","Habitat"),new=c("FP_Areakm2","Floods_1941","G
roundwater_1941","Sediment_1941","Organics.Solutes_1941","Habitat_1941"))

#1945
function.data.MRB <- merge(function.data.MRB, IFI_Scaled1945[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_1945","Groundwater_1945","Sediment_19
45","Organics.Solutes_1945","Habitat_1945"))
#1950
function.data.MRB <- merge(function.data.MRB, IFI_Scaled1950[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_1950","Groundwater_1950","Sediment_19
50","Organics.Solutes_1950","Habitat_1950"))
#1955
function.data.MRB <- merge(function.data.MRB, IFI_Scaled1955[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_1955","Groundwater_1955","Sediment_19
55","Organics.Solutes_1955","Habitat_1955"))
#1960
function.data.MRB <- merge(function.data.MRB, IFI_Scaled1960[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_1960","Groundwater_1960","Sediment_19
60","Organics.Solutes_1960","Habitat_1960"))
#1965
function.data.MRB <- merge(function.data.MRB, IFI_Scaled1965[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_1965","Groundwater_1965","Sediment_19
65","Organics.Solutes_1965","Habitat_1965"))
#1970
function.data.MRB <- merge(function.data.MRB, IFI_Scaled1970[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_1970","Groundwater_1970","Sediment_19
70","Organics.Solutes_1970","Habitat_1970"))
#1975
```

```
function.data.MRB <- merge(function.data.MRB, IFI_Scaled1975[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_1975","Groundwater_1975","Sediment_19
75","Organics.Solutes_1975","Habitat_1975"))
#1980
function.data.MRB <- merge(function.data.MRB, IFI_Scaled1980[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_1980","Groundwater_1980","Sediment_19
80","Organics.Solutes_1980","Habitat_1980"))
#1985
function.data.MRB <- merge(function.data.MRB, IFI_Scaled1985[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_1985","Groundwater_1985","Sediment_19
85","Organics.Solutes_1985","Habitat_1985"))
#1990
function.data.MRB <- merge(function.data.MRB, IFI_Scaled1990[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_1990","Groundwater_1990","Sediment_19
90","Organics.Solutes_1990","Habitat_1990"))
#1995
function.data.MRB <- merge(function.data.MRB, IFI_Scaled1995[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_1995","Groundwater_1995","Sediment_19
95","Organics.Solutes_1995","Habitat_1995"))
#2000
function.data.MRB <- merge(function.data.MRB, IFI_Scaled2000[,c(2,(3:7))],
by = "HUC12", all.x = TRUE)
setnames(function.data.MRB,old=c("Floods","Groundwater","Sediment","Organi
cs.Solutes","Habitat"),new=c("Floods_2000","Groundwater_2000","Sediment_20
00","Organics.Solutes_2000","Habitat_2000"))

#write.csv(function.data.MRB,paste(out.path,"function.IFI.MRB.csv"))

#floods
Floods <- function.data.MRB[,c(5,10,15,20,25,30,35,40,45,50,55,60,65)]
#Groundwater
Groundwater <-
function.data.MRB[,c(6,11,16,21,26,31,36,41,46,51,56,61,66)]
#sediment
Sediment <- function.data.MRB[,c(7,12,17,22,27,32,37,42,47,52,57,62,67)]
#organics solutes
Organics.Solutes <-
function.data.MRB[,c(8,13,18,23,28,33,38,43,48,53,58,63,68)]
#habitat
Habitat <- function.data.MRB[,c(9,14,19,24,29,34,39,44,49,54,59,64,69)]


Floods_comparisons <-
list(c("Floods_1941","Floods_1945"),c("Floods_1945","Floods_1950"),c("Floo
```

```
ds_1950","Floods_1955"),c("Floods_1955","Floods_1960"),c("Floods_1960","Fl
oods_1965"),c("Floods_1965","Floods_1970"),c("Floods_1970","Floods_1975"),
c("Floods_1975","Floods_1980"),c("Floods_1980","Floods_1985"),c("Floods_19
85","Floods_1990"),c("Floods_1990","Floods_1995"),c("Floods_1995","Floods_
2000"))

pFloods <- ggplot(melt(Floods),aes(variable, value,fill=variable)) +
  geom_boxplot(alpha=0.5)+
  #stat_summary(fun=mean, geom="point", shape=8, size=3, color="grey0",
fill="grey0")+
  #stat_summary(fun.data = fun_median, geom="text",size=3,vjust=1)+
  theme(legend.position="none")+
  labs(x="Year",y="Flood Reduction IFI",col='') +
  scale_x_discrete(labels =
c("1941","1945","1950","1955","1960","1965","1970","1975","1980","1985","1
990","1995","2000"))+
  scale_fill_manual(values=as.vector(ocean.delta(13)))+
  stat_compare_means(comparisons = Floods_comparisons, label = "p.signif",
method = "t.test",
                     label.y = seq(1, 1.2,
0.2/length(Floods_comparisons)), size = 3)+
  theme(panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank())



#plot(pFloods)

Groundwater_comparisons <-
list(c("Groundwater_1941","Groundwater_1945"),c("Groundwater_1945","Ground
water_1950"),c("Groundwater_1950","Groundwater_1955"),c("Groundwater_1955"
,"Groundwater_1960"),c("Groundwater_1960","Groundwater_1965"),c("Groundwat
er_1965","Groundwater_1970"),c("Groundwater_1970","Groundwater_1975"),c("G
roundwater_1975","Groundwater_1980"),c("Groundwater_1980","Groundwater_198
5"),c("Groundwater_1985","Groundwater_1990"),c("Groundwater_1990","Groundw
ater_1995"),c("Groundwater_1995","Groundwater_2000"))

pGroundwater <- ggplot(melt(Groundwater),aes(variable,
value,fill=variable)) +
  geom_boxplot(alpha=0.5)+
  #stat_summary(fun=mean, geom="point", shape=8, size=3, color="grey0",
fill="grey0")+
  #stat_summary(fun.data = fun_median, geom="text",size=3,vjust=1)+
  theme(legend.position="none")+
  labs(x="Year",y="Grounwater Storage IFI",col='') +
  scale_x_discrete(labels =
c("1941","1945","1950","1955","1960","1965","1970","1975","1980","1985","1
990","1995","2000"))+
  scale_fill_manual(values=as.vector(ocean.delta(13)))+
  stat_compare_means(comparisons = Groundwater_comparisons, label =
"p.signif", method = "t.test",
```

```
                    label.y = seq(1, 1.2,
0.2/length(Groundwater_comparisons)), size = 3)+
  theme( panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank())


#plot(pGroundwater)


Sediment_comparisons <-
list(c("Sediment_1941","Sediment_1945"),c("Sediment_1945","Sediment_1950")
,c("Sediment_1950","Sediment_1955"),c("Sediment_1955","Sediment_1960"),c("
Sediment_1960","Sediment_1965"),c("Sediment_1965","Sediment_1970"),c("Sedi
ment_1970","Sediment_1975"),c("Sediment_1975","Sediment_1980"),c("Sediment
_1980","Sediment_1985"),c("Sediment_1985","Sediment_1990"),c("Sediment_199
0","Sediment_1995"),c("Sediment_1995","Sediment_2000"))

pSediment <- ggplot(melt(Sediment),aes(variable, value,fill=variable)) +
  geom_boxplot(alpha=0.5)+
  #stat_summary(fun=mean, geom="point", shape=8, size=3, color="grey0",
fill="grey0")+
  #stat_summary(fun.data = fun_median, geom="text",size=3,vjust=1)+
  theme(legend.position="none")+
  labs(x="Year",y="Sediment Regulation IFI",col='') +
  scale_x_discrete(labels =
c("1941","1945","1950","1955","1960","1965","1970","1975","1980","1985","1
990","1995","2000"))+
  scale_fill_manual(values=as.vector(ocean.delta(13)))+
  stat_compare_means(comparisons = Sediment_comparisons, label =
"p.signif", method = "t.test",
                    label.y = seq(1, 1.2,
0.2/length(Sediment_comparisons)), size = 3)+
  theme(panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank())


#plot(pSediment)

Organics.Solutes_comparisons <-
list(c("Organics.Solutes_1941","Organics.Solutes_1945"),c("Organics.Solute
s_1945","Organics.Solutes_1950"),c("Organics.Solutes_1950","Organics.Solut
es_1955"),c("Organics.Solutes_1955","Organics.Solutes_1960"),c("Organics.S
olutes_1960","Organics.Solutes_1965"),c("Organics.Solutes_1965","Organics.
Solutes_1970"),c("Organics.Solutes_1970","Organics.Solutes_1975"),c("Organ
ics.Solutes_1975","Organics.Solutes_1980"),c("Organics.Solutes_1980","Orga
nics.Solutes_1985"),c("Organics.Solutes_1985","Organics.Solutes_1990"),c("
Organics.Solutes_1990","Organics.Solutes_1995"),c("Organics.Solutes_1995",
"Organics.Solutes_2000"))

pOrganics.Solutes <- ggplot(melt(Organics.Solutes),aes(variable,
value,fill=variable)) +
  geom_boxplot(alpha=0.5)+
```

```
  #stat_summary(fun=mean, geom="point", shape=8, size=3, color="grey0",
fill="grey0")+
  #stat_summary(fun.data = fun_median, geom="text",size=3,vjust=1)+
  theme(legend.position="none")+
  labs(x="Year",y="Organics/Solutes Regulation IFI",col='') +
  scale_x_discrete(labels =
c("1941","1945","1950","1955","1960","1965","1970","1975","1980","1985","1
990","1995","2000"))+
  scale_fill_manual(values=as.vector(ocean.delta(13)))+
  stat_compare_means(comparisons = Organics.Solutes_comparisons, label =
"p.signif", method = "t.test",
                     label.y = seq(1, 1.2,
0.2/length(Organics.Solutes_comparisons)), size = 3)+
  theme(panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank())


#plot(pOrganics.Solutes)

Habitat_comparisons <-
list(c("Habitat_1941","Habitat_1945"),c("Habitat_1945","Habitat_1950"),c("
Habitat_1950","Habitat_1955"),c("Habitat_1955","Habitat_1960"),c("Habitat_
1960","Habitat_1965"),c("Habitat_1965","Habitat_1970"),c("Habitat_1970","H
abitat_1975"),c("Habitat_1975","Habitat_1980"),c("Habitat_1980","Habitat_1
985"),c("Habitat_1985","Habitat_1990"),c("Habitat_1990","Habitat_1995"),c(
"Habitat_1995","Habitat_2000"))

pHabitat <- ggplot(melt(Habitat),aes(variable, value,fill=variable)) +
  geom_boxplot(alpha=0.5)+
  #stat_summary(fun=mean, geom="point", shape=8, size=3, color="grey0",
fill="grey0")+
  #stat_summary(fun.data = fun_median, geom="text",size=3,vjust=1)+
  theme(legend.position="none")+
  labs(x="Year",y="Habitat Provisioning IFI",col='') +
  scale_x_discrete(labels =
c("1941","1945","1950","1955","1960","1965","1970","1975","1980","1985","1
990","1995","2000"))+
  scale_fill_manual(values=as.vector(ocean.delta(13)))+
  stat_compare_means(comparisons = Habitat_comparisons, label =
"p.signif", method = "t.test",
                     label.y = seq(1, 1.2,
0.2/length(Habitat_comparisons)), size = 3)+
  theme(panel.grid.major.x =
          element_blank(), panel.grid.major.y = element_blank(),
panel.grid.minor.y = element_blank())


#plot(pHabitat)

# save plots
function.IFI.plots <-grid.arrange(pFloods,
pGroundwater,pSediment,pOrganics.Solutes,pHabitat,p0, ncol = 3,nrow=2)
```

```
#ggsave("all IFI by year boxplot.png", plot = function.IFI.plots, path =
out.path, width = 16, height = 5.5, units = "in", dpi = 1000)

ggsave("all IFI by year boxplot.pdf", plot = function.IFI.plots, path =
out.path, width = 16, height = 5.5, units = "in", dpi = 300)



############################

# some statistical summaries by year

years <-
list("1941","1945","1950","1955","1960","1965","1970","1975","1980","1985"
,"1990","1995","2000")
year <-as.numeric(years)
IFI.Summary.year <-data.frame(year)
IFI.Summary.year$MeanIFI <- t(IFI.by.year %>% summarise_if(is.numeric,
mean))
IFI.Summary.year$tenth_IFI <- t(IFI.by.year %>% summarise_if(is.numeric,
~quantile(.x,probs = 0.10)))
IFI.Summary.year$ninety_IFI <- t(IFI.by.year %>% summarise_if(is.numeric,
~quantile(.x,probs = 0.90)))
IFI.Summary.year$MedianIFI <- t(IFI.by.year %>% summarise_if(is.numeric,
median))



p1 <- ggplot(IFI.Summary.year,aes(x=year))+
geom_point(aes(y=MeanIFI,color="IFI Mean"))+
  geom_point(aes(y=MedianIFI,color= "IFI Median"))+
  geom_point(aes(y=tenth_IFI,color="IFI 10th Percentile")) +
  geom_point(aes(y=ninety_IFI,color= "IFI 90th Percentile"))+
  labs(x="Year",y="Overall IFI",col='')

plot(p1)


# # save plot
# out.graph <- paste(out.path, "IFI compare.png", sep="")
# png(filename =out.graph, width = 500, height = 400,res=85)
# p1
# dev.off()

#looking at change in AG and Developed percentage over time

densityinfo <-
read.csv(paste(basepath,"\\MergedOutputs\\Combined_Data_MRB.csv",sep=""))


IFI.Summary.year$MeanDeveloped <- t(densityinfo[,c(5:17)] %>%
summarise_if(is.numeric, mean))
```

```
IFI.Summary.year$MeanAg <- t(densityinfo[,c(18:30)] %>%
summarise_if(is.numeric, mean))

p2 <- ggplot(IFI.Summary.year,aes(x=year))+
geom_point(aes(y=MeanDeveloped,color="Mean Developed %"))+
  geom_point(aes(y=MeanAg,color="Mean Ag %")) +
  labs(x="Year",y="Percent of Floodplain Area",col='')

plot(p2)
#
# # save plot
# out.graph <- paste(out.path, "Ag and Developed compare.png", sep="")
# png(filename =out.graph, width = 500, height = 400,res=85)
# p2
# dev.off()


years <-
list("1941","1945","1950","1955","1960","1965","1970","1975","1980","1985"
,"1990","1995","2000")
year <-as.numeric(years)
function.Summary.year <-data.frame(year)
#floods
function.Summary.year$Floods <-
t(function.data.MRB[,c(5,10,15,20,25,30,35,40,45,50,55,60,65)] %>%
summarise_if(is.numeric, mean))
#Groundwater
function.Summary.year$Groundwater <-
t(function.data.MRB[,c(6,11,16,21,26,31,36,41,46,51,56,61,66)] %>%
summarise_if(is.numeric, mean))
#sediment
function.Summary.year$Sediment <-
t(function.data.MRB[,c(7,12,17,22,27,32,37,42,47,52,57,62,67)] %>%
summarise_if(is.numeric, mean))
#organics solutes
function.Summary.year$Organics_Solutes <-
t(function.data.MRB[,c(8,13,18,23,28,33,38,43,48,53,58,63,68)] %>%
summarise_if(is.numeric, mean))
#habitat
function.Summary.year$Habitat <-
t(function.data.MRB[,c(9,14,19,24,29,34,39,44,49,54,59,64,69)] %>%
summarise_if(is.numeric, mean))

#create plot
p3 <- ggplot(function.Summary.year,aes(x=year))+
geom_point(aes(y=Floods,color="Mean IFI Floods"))+
  geom_point(aes(y=Groundwater,color="Mean IFI Groundwater")) +
  geom_point(aes(y=Sediment,color="Mean IFI Sediment")) +
  geom_point(aes(y=Organics_Solutes,color="Mean IFI Organics/Solutes")) +
  geom_point(aes(y=Habitat,color="Mean IFI Habitat")) +
  labs(x="Year",y="Function IFI",col='')

plot(p3)
```

```
# # save plot
# out.graph <- paste(out.path, "Function IFI compare.png", sep="")
# png(filename =out.graph, width = 500, height = 400,res=85)
# p3
# dev.off()

####
```

IFI mapping

The following R scripts were used to map the IFI dataset.

```
#IFI mapping for entire US

pacman::p_load(ggplot2,RColorBrewer,wesanderson,tidyr,grid,gridExtra,dplyr
,data.table,stringr,tidyverse,tools,reshape2,broom,rgdal,ggmap,cowplot,map
s,mapdata)

#paths and IFI data

# set paths
basepath <- "K:\\Kira Simonson\\Stressor Densities"
out.path <- paste(basepath, "\\Results\\", sep="")
out.path.thesis <-"K:\\Kira Simonson\\Thesis\\Figures\\"

#get floodplains shapefile

#get IFI data
IFI.data <- read.csv("K:\\Kira Simonson\\Stressor Densities\\Results\\
IFI.stats.csv")


cuts_labels <- c("less than 0.3","0.3-0.4","0.4-0.5","0.5-0.6","0.6-
0.7","0.7-0.8","0.8-0.9","0.9-1.0")
IFI.data$cuts <- cut(IFI.data$IFI_geomean,
                     breaks = c(0,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1),labels =
as.character(cuts_labels))

HUC12_W <- readOGR("K:\\Kira Simonson\\Stressor
Densities\\Shapefiles\\W_US_HUC12.shp")
HUC12_W_tidy <- tidy(HUC12_W,region='huc12')


#get tidied data
HUC12_E_tidy<-read.csv("K:\\Kira Simonson\\Stressor
Densities\\Shapefiles\\ HUC12_E_tidy.stats.csv")


IFI.data$HUC12 <- as.numeric(IFI.data$HUC12)



#join with IFI data by HUC12

HUC12_E_map <- inner_join(HUC12_E_tidy,IFI.data, by=c("id"="HUC12"))

HUC12_W_tidy$id <- as.numeric(HUC12_W_tidy$id)
HUC12_W_map <- inner_join(HUC12_W_tidy,IFI.data,  by=c("id"="HUC12"))
```

```
maphuc12_W <- ggplot() + geom_polygon(data = HUC12_W_map, aes(x = long, y
= lat, group = group, fill = cuts))+
  coord_map()+
  scale_fill_manual(values = c('#671733','#913a3e','#C16F4A', '#DDB96C'
      ,'#F5EDA2','#C3E09B', '#91B892'  ,'#5575A8','#493a87')) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "Overall IFI")


#ggsave("IFI W map (recover).pdf", plot = maphuc12_W, path = out.path,
width = 6.5, height = 5, units = "in", dpi = 72)



state <- map_data("state")
map_US <- maphuc12_W+
  geom_polygon(data = HUC12_E_map, aes(x = long, y = lat, group = group,
fill = cuts))+
  coord_map()+
  scale_fill_manual(values = c('#671733','#913a3e','#C16F4A', '#DDB96C'
      ,'#F5EDA2','#C3E09B', '#91B892'  ,'#5575A8','#493a87')) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "Overall IFI")+
  geom_polygon(data=state, aes(x=long,
y=lat,group=group),color="black",size=0.01, fill=NA)+coord_map()


#ggsave("IFI US map (recover).pdf", plot = map_US, path = out.path, width
= 6.5, height = 7.5, units = "in", dpi = 500)
ggsave("IFI US map (recover).png", plot = map_US, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)



huc4s <- readOGR("K:\\Kira Simonson\\Stressor
Densities\\Shapefiles\\selected_huc4s.shp")
huc4s_tidy <- tidy(huc4s, region="huc4")

map_US_HUC4 <- map_US + geom_polygon(data=huc4s_tidy, aes(x=long,
y=lat,group=group),color="red",size=0.5, fill=NA)+coord_map()
```

```
ggsave("IFI US map (regions).pdf", plot = map_US_HUC4, path = out.path,
width = 6.5, height = 7.5, units = "in", dpi = 72)

#IFI Mapping MRB
pacman::p_load(ggplot2,RColorBrewer,wesanderson,tidyr,grid,gridExtra,dplyr
,data.table,stringr,tidyverse,tools,reshape2,broom,rgdal,ggmap,cowplot,pal
s,EnvStats,maps,mapdata)

#paths and IFI data

# set paths
basepath <- "K:\\Kira Simonson\\Stressor Densities\\MRB"
out.path <- paste(basepath, "\\Results\\", sep="")
out.path.thesis.MRB <- "K:\\Kira Simonson\\Thesis\\Figures\\Appendix
MRB\\"

#get floodplains shapefile

HUC12 <- readOGR("K:\\Kira Simonson\\Stressor Densities\\MRB\\MRB
Shapefile\\HUC12_MRB.shp")

HUC2 <- readOGR("K:\\Kira Simonson\\Stressor Densities\\MRB\\MRB
Shapefile\\HUC2_MRB.shp")

HUC4 <- readOGR("K:\\Kira Simonson\\Stressor Densities\\MRB\\MRB
Shapefile\\HUC4_MRB.shp")

HUC6 <- readOGR("K:\\Kira Simonson\\Stressor Densities\\MRB\\MRB
Shapefile\\HUC6_MRB.shp")
#MRBfloodplains <- readOGR("K:\\Kira Simonson\\Stressor
Densities\\MRB\\MRB Shapefile\\MRBfloodplains.shp")

#get IFI data
IFI.data <- read.csv("K:\\Kira Simonson\\Stressor
Densities\\MRB\\Results\\ all.IFI.MRB.csv")
IFI.data[1:14661,"HUC12"] <- paste("0",IFI.data[1:14661,"HUC12"],sep="")

IFI.data$HUC2 <- substring(IFI.data$HUC12,1,2)
IFI.data$HUC2 <- as.character(IFI.data$HUC2)


#IFI by function data
function.data <- read.csv("K:\\Kira Simonson\\Stressor
Densities\\MRB\\Results\\ function.IFI.MRB.csv")

#read in all IFI data by year
IFI.1941 <-IFI.data[,c("HUC2","HUC12","IFI_1941")]
IFI.1945 <-IFI.data[,c("HUC2","HUC12","IFI_1945")]
IFI.1950 <-IFI.data[,c("HUC2","HUC12","IFI_1950")]
IFI.1955 <-IFI.data[,c("HUC2","HUC12","IFI_1955")]
IFI.1960 <-IFI.data[,c("HUC2","HUC12","IFI_1960")]
IFI.1965 <-IFI.data[,c("HUC2","HUC12","IFI_1965")]
IFI.1970 <-IFI.data[,c("HUC2","HUC12","IFI_1970")]
IFI.1975 <-IFI.data[,c("HUC2","HUC12","IFI_1975")]
```

```
IFI.1980 <-IFI.data[,c("HUC2","HUC12","IFI_1980")]

IFI.1985 <-IFI.data[,c("HUC2","HUC12","IFI_1985")]
IFI.1990 <-IFI.data[,c("HUC2","HUC12","IFI_1990")]
IFI.1995 <-IFI.data[,c("HUC2","HUC12","IFI_1995")]
IFI.2000 <-IFI.data[,c("HUC2","HUC12","IFI_2000")]


# transform for ggplot
HUC12_tidy <- tidy(HUC12, region = "huc12")

#join with IFI data
HUC12_map_1941 <- left_join(HUC12_tidy,IFI.1941, by = c("id" = "HUC12"))
HUC12_map_1945 <- left_join(HUC12_tidy,IFI.1945, by = c("id" = "HUC12"))
HUC12_map_1950 <- left_join(HUC12_tidy,IFI.1950, by = c("id" = "HUC12"))
HUC12_map_1955 <- left_join(HUC12_tidy,IFI.1955, by = c("id" = "HUC12"))
HUC12_map_1960 <- left_join(HUC12_tidy,IFI.1960, by = c("id" = "HUC12"))
HUC12_map_1965 <- left_join(HUC12_tidy,IFI.1965, by = c("id" = "HUC12"))
HUC12_map_1970 <- left_join(HUC12_tidy,IFI.1970, by = c("id" = "HUC12"))
HUC12_map_1975 <- left_join(HUC12_tidy,IFI.1975, by = c("id" = "HUC12"))
HUC12_map_1980 <- left_join(HUC12_tidy,IFI.1980, by = c("id" = "HUC12"))
HUC12_map_1985 <- left_join(HUC12_tidy,IFI.1985, by = c("id" = "HUC12"))
HUC12_map_1990 <- left_join(HUC12_tidy,IFI.1990, by = c("id" = "HUC12"))
HUC12_map_1995 <- left_join(HUC12_tidy,IFI.1995, by = c("id" = "HUC12"))
HUC12_map_2000 <- left_join(HUC12_tidy,IFI.2000, by = c("id" = "HUC12"))

# Combined tidied data with summary statstics for each HUC-12
map1941 <- ggplot() + geom_polygon(data = HUC12_map_1941, aes(x = long, y
= lat, group = group, fill = IFI_1941))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1941")

map1945 <- ggplot() + geom_polygon(data = HUC12_map_1945, aes(x = long, y
= lat, group = group, fill = IFI_1945))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
```

```
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1945")

map1950 <- ggplot() + geom_polygon(data = HUC12_map_1950, aes(x = long, y
= lat, group = group, fill = IFI_1950))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1950")

map1955 <- ggplot() + geom_polygon(data = HUC12_map_1955, aes(x = long, y
= lat, group = group, fill = IFI_1955))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1955")

map1960 <- ggplot() + geom_polygon(data = HUC12_map_1960, aes(x = long, y
= lat, group = group, fill = IFI_1960))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1960")
```

```
map1965 <- ggplot() + geom_polygon(data = HUC12_map_1965, aes(x = long, y
= lat, group = group, fill = IFI_1965))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1965")

map1970 <- ggplot() + geom_polygon(data = HUC12_map_1970, aes(x = long, y
= lat, group = group, fill = IFI_1970))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1970")

map1975 <- ggplot() + geom_polygon(data = HUC12_map_1975, aes(x = long, y
= lat, group = group, fill = IFI_1975))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1975")

map1980 <- ggplot() + geom_polygon(data = HUC12_map_1980, aes(x = long, y
= lat, group = group, fill = IFI_1980))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
```

```
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1980")

map1985 <- ggplot() + geom_polygon(data = HUC12_map_1985, aes(x = long, y
= lat, group = group, fill = IFI_1985))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1985")

map1990 <- ggplot() + geom_polygon(data = HUC12_map_1990, aes(x = long, y
= lat, group = group, fill = IFI_1990))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1990")

map1995 <- ggplot() + geom_polygon(data = HUC12_map_1995, aes(x = long, y
= lat, group = group, fill = IFI_1995))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
```

```
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "1995")

map2000 <- ggplot() + geom_polygon(data = HUC12_map_2000, aes(x = long, y
= lat, group = group, fill = IFI_2000))+
  coord_equal() +
  scale_fill_gradientn(colours =
c('sienna','sienna1','papayawhip','cadetblue1','cornflowerblue'), breaks =
seq(0, 1.0, by = 0.2),labels = c("0.0","0.2", "0.4", "0.6", "0.8","1.0"),
limits = c(0,1)) +
  labs(x = NULL, y = NULL, fill = "IFI")+
  theme_minimal() +
  theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())+
  labs(tag = "2000")


# # # save plot
# ggsave("map huc 12 1941.pdf", plot = map1941, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 1945.pdf", plot = map1945, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 1950.pdf", plot = map1950, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 1955.pdf", plot = map1955, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 1960.pdf", plot = map1960, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 1965.pdf", plot = map1965, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 1970.pdf", plot = map1970, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 1975.pdf", plot = map1975, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 1980.pdf", plot = map1980, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 1985.pdf", plot = map1985, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 1990.pdf", plot = map1990, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 1995.pdf", plot = map1995, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)
# ggsave("map huc 12 2000.pdf", plot = map2000, path = out.path, width =
6.5, height = 7.5, units = "in", dpi = 1000)


ggsave("map huc 12 1941.png", plot = map1941, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
```

```
ggsave("map huc 12 1945.png", plot = map1945, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
ggsave("map huc 12 1950.png", plot = map1950, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
ggsave("map huc 12 1955.png", plot = map1955, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
ggsave("map huc 12 1960.png", plot = map1960, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
ggsave("map huc 12 1965.png", plot = map1965, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
ggsave("map huc 12 1970.png", plot = map1970, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
ggsave("map huc 12 1975.png", plot = map1975, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
ggsave("map huc 12 1980.png", plot = map1980, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
ggsave("map huc 12 1985.png", plot = map1985, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
ggsave("map huc 12 1990.png", plot = map1990, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
ggsave("map huc 12 1995.png", plot = map1995, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)
ggsave("map huc 12 2000.png", plot = map2000, path = out.path.thesis.MRB,
width = 6.5, height = 7.5, units = "in", dpi = 1000)


# save plots
#IFI.maps <-
grid.arrange(map1941,map1945,map1950,map1955,map1960,map1965,map1970,map19
75,map1980,map1985,map1990,map1995,map2000,ncol = 4,nrow=4)


#ggsave("maps by year.pdf", plot = IFI.maps, path = out.path, width = 16,
height = 5.5, units = "in", dpi = 1000)

# map showing degree of change from 1941 to 2000 and change vs no change
#negative or positive change from 1941 to 2000?
IFI.change <- IFI.data[,c("HUC12","IFI_1941","IFI_2000")]
IFI.change$percentchange <- ((IFI.change$IFI_2000-
IFI.change$IFI_1941)/(IFI.change$IFI_1941))*100

# finding a density curve of change




densityplot <-
  IFI.change %>%
  ggplot( aes(x=percentchange)) +
  geom_density(fill="sienna3", color="sienna3", alpha=0.8)+
  xlim(-10,10)+
  xlab("Percent Change in IFI from 1941 to 2000") +
  ylab("Density") +
  theme_bw() +
```

```
  theme(text = element_text(size=11))

densityplot

ggsave("density plot IFI percent
change.pdf",plot=densityplot,path=out.path,width = 4, height = 4, units =
"in", dpi = 72)



# direction?
IFI.change$direction <-
as.numeric(ifelse(IFI.change$percentchange==0,0,as.numeric(ifelse(IFI.chan
ge$percentchange>0,1,-1))))


#IFI.change$cuts <- cut(IFI.change$percentchange,
                  # breaks = c(Inf,.1,.05,.01,.001,0,-.001, -.01, -.05, -
.1, -Inf))
                  #  labels = c('0.1-100','0.05-0.1','0.01-0.05','0.001-
0.01','0','0.01-0.001','.01-0.05','0.05-0.01','0.1-0.01','100-0.1')


#join data spatially to map
HUC12_map_change <- inner_join(HUC12_tidy,IFI.change, by = c("id" =
"HUC12"))


#
# mapchange1 <- ggplot() + geom_polygon(data = HUC12_map_change, aes(x =
long, y = lat, group = group, fill = cuts))+
#   coord_map() +
#   scale_fill_manual(values=as.vector(ocean.balance(18)),direction=-1) +
#   labs(x = NULL, y = NULL, fill = "IFI Change 1941-2000")+
#   theme_minimal() +
#   theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
#   theme(axis.text=element_blank()) +
#   theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())
#
# ggsave("map change 1941-2000.pdf", plot = mapchange1, path = out.path,
width = 6.5, height = 7.5, units = "in", dpi = 1000)

usa <- map_data("usa")

mapchange2 <- ggplot() + geom_polygon(data = HUC12_map_change, aes(x =
long, y = lat, group = group, fill = direction))+
  geom_polygon(data=usa, aes(x=long,
y=lat,group=group),color="black",size=0.25, fill=NA)+
  coord_map() +
  scale_fill_gradientn(colours = c("darkred", "papayawhip","darkcyan"),
breaks = seq(-1.0, 1.0, by = 1),labels = c("Decrease in IFI","No Change",
"Increase in IFI"), limits = c(-1,1)) +
```

```
  labs(x = NULL, y = NULL, fill = "IFI Change 1941-2000")+
  theme_minimal() +
  theme(legend.position = "none")+
  #theme(legend.text = element_text(size = 10), legend.key.width =
unit(0.75, "cm")) +
  theme(axis.text=element_blank()) +
  theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())


ggsave("map change 1941-2000 (2).pdf", plot = mapchange2, path = out.path,
width = 6.5, height = 7.5, units = "in", dpi = 72)


trajectory <- ggarrange(mapchange2,densityplot, widths = c(2, 0.7),
          ncol = 2, nrow = 1)

ggsave("trajectory map 2.png", plot = trajectory, path = out.path.thesis,
width = 6.5, height = 4, units = "in", dpi = 1000)


#
# US <- readOGR("K:\\Kira Simonson\\National Data Downloads\\US
boundary\\cb_2018_us_nation_20m.shp")
# MRBpolygon <- readOGR("K:\\Kira Simonson\\National Data Downloads\\SP
22\\Mississippi River Basin\\Input Data\\MRB_Boundary.shp")
#
# US_tidy <- tidy(US,region="NAME")
# MRB_tidy <- tidy(MRBpolygon, region="MRB")
#
# MRB_reference <- ggplot()+
#   geom_polygon(data = US_tidy, aes(x = long, y = lat, group = group),
color = "grey8") +
#   #geom_polygon(data = MRB_tidy, aes(x = long, y = lat, group = group),
fill = "darkseagreen") +
#   coord_equal() +
#   labs(x = NULL, y = NULL) +
#   theme_minimal(base_size = 14) +
#   theme(legend.text = element_text(size = 8)) +
#   theme(axis.text=element_blank()) +
#   theme(panel.grid.major = element_blank(), panel.grid.minor =
element_blank())
#
```